
SSSSSSSSSSS 000000000 RRRRRRRRRRR TTTTTTTTTTTTTTT 333333333 222222222
SSSSSSSSSSS 000000000 RRRRRRRRRRR TTTTTTTTTTTTTTT 333333333 222222222
SSSSSSSSSSS 000000000 RRRRRRRRRRR TTTTTTTTTTTTTTT 333333333 222222222
SSS 000 000 RRR RRR TTT 333 333 222 222
SSS 000 000 RRR RRR TTT 333 333 222 222
SSS 000 000 RRR RRR TTT 333 333 222 222
SSS 000 000 RRR RRR TTT 333 222
SSS 000 000 RRR RRR TTT 333 222
SSSSSSSSS 000 000 RRRRRRRRRRR TTT 333 222
SSSSSSSSS 000 000 RRRRRRRRRRR TTT 333 222
SSSSSSSSS 000 000 RRRRRRRRRRR TTT 333 222
SSS 000 000 RRR RRR TTT 333 222
SSS 000 000 RRR RRR TTT 333 222
SSS 000 000 RRR RRR TTT 333 222
SSS 000 000 RRR RRR TTT 333 333 222
SSS 000 000 RRR RRR TTT 333 333 222
SSS 000 000 RRR RRR TTT 333 333 222
SSSSSSSSSSS 000000000 RRR RRR TTT 333333333 222222222222222
SSSSSSSSSSS 000000000 RRR RRR TTT 333333333 222222222222222
SSSSSSSSSSS 000000000 RRR RRR TTT 333333333 222222222222222

FILEID**SORSPEC

SOR
V04

```
;     1          0001  0 MODULE SOR$SPEC_FILE (
;     2          0002  0                 IDENT = 'V04-000'       ! File: SORSPEC.B32 Edit: PDG3030
;     3          0003  0                 ) =
;     4          0004  1 BEGIN
;     5          0005  1 !
;     6          0006  1 !**********************************************************************
;     7          0007  1 !*                                                                    *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                          *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.           *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                           *
;    11          0011  1 !*                                                                    *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
;    13          0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
;    17          0017  1 !*  TRANSFERRED.                                                       *
;    18          0018  1 !*                                                                    *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
;    20          0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
;    21          0021  1 !*  CORPORATION.                                                       *
;    22          0022  1 !*                                                                    *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.            *
;    25          0025  1 !*                                                                    *
;    26          0026  1 !*                                                                    *
;    27          0027  1 !**********************************************************************
;    28          0028  1 !
;    29          0029  1
;    30          0030  1 !++
;    31          0031  1 !
;    32          0032  1 ! FACILITY:     VAX-11 SORT/MERGE
;    33          0033  1 !
;    34          0034  1 ! ABSTRACT:
;    35          0035  1 !
;    36          0036  1 !       This module contains routines that read and process specification text.
;    37          0037  1 !
;    38          0038  1 ! ENVIRONMENT:  VAX/VMS user mode
;    39          0039  1 !
;    40          0040  1 ! AUTHOR: Peter D Gilbert, CREATION DATE: 07-Jan-1982
;    41          0041  1 !
;    42          0042  1 ! MODIFIED BY:
;    43          0043  1 !
;    44          0044  1 !       T03-015         Original
;    45          0045  1 !       T03-016 Copy relevant information to RDT entries with same KFT indices.
;    46          0046  1 !               Improve calculation of COM_FORMATS.  Comments.  PDG 13-Dec-1982
;    47          0047  1 !       T03-017 Put a linkage declaration on SOR$$COMPARE.  PDG-15-Dec-1982
;    48          0048  1 !       T03-018 Define offsets for use by SOR$$COMPARE.  PDG 22-Dec-1982
;    49          0049  1 !       T03-019 Check for a longword temporary (not CTX[COM_LRL_INT) exceeding
;    50          0050  1 !               MAX_REFSIZE.  PDG 28-Dec-1982
;    51          0051  1 !       T03-020 Added the output format record length as an output parameter
;    52          0052  1 !               from SOR$$REFORM.  PDG 3-Jan-1983
;    53          0053  1 !       T03-021 Added clean-up routine for the work area.  PDG 26-Jan-1983
;    54          0054  1 !       T03-022 Use COM_MRG_STREAM for stable merges.  PDG 27-Jan-1983
;    55          0055  1 !       T03-023 Define COM$B_PAD for use by SOR$$COMPARE.  PDG 8-Feb-1983
;    56          0056  1 !       T03-024 Abort on errors from SOR$$SFPRS.  Use KFT_NDE_SIZ.
;    57          0057  1 !               Pass the context address to callback routines.  PDG 12-Feb-1983
```

```
;    58          0058  1 !       T03-025 Use SOR$$(DE)ALLOCATE to append code strings.  PDG 7-Mar-1983
;    59          0059  1 !       T03-026 Special-case some stuff to use SOR$$KEY_SUB.  PDG 17-Mar-1983
;    60          0060  1 !       T03-027 Correctly set the COM_VAR flag.  PDG 9-May-1983
;    61          0061  1 !       T03-028 Fix adding DSC_ADR to COM_COMPARE.  Make allowances for ADDRESS
;    62          0062  1 !               and INDEX sorts.  PDG 10-May-1983
;    63          0063  1 !       T03-029 Leave COM_EQUAL equal to 0 if it's not needed.  PDG 26-Aug-1983
;    64          0064  1 !       T03-030 SOR$$BEST_FILE_NAME assumes NAM$B_RSL and NAM$B_ESL are zero
;    65          0065  1 !               before the OPEN or CREATE.  PDG 10-Nov-1983
;    66          0066  1 !--
```

```
;    68          0067  1 LIBRARY 'SYS$LIBRARY:STARLET';
;    69          0068  1 REQUIRE 'SRC$:COM';
;    70          0138  1 LIBRARY 'SRC$:SRTSPC';
;    71          0139  1 LIBRARY 'SRC$:OPCODES';
;    72          0140  1
;    73          0141  1 !%IF %DECLARED(%QUOTE $DESCRIPTOR) %THEN UNDECLARE %QUOTE $DESCRIPTOR; %FI
;    74          0142  1
;    75          0143  1 FORWARD ROUTINE
;    76          0144  1     SOR$$SPEC_FILE:     CAL_CTXREG,                 ! Process specification text
;    77          0145  1     CALC_LRL_OUT:       CAL_CTXREG NOVALUE,         ! Spec file processing for LRL
;    78          0146  1     SOR$$SPEC_KEY_SUB:  CAL_CTXREG,                 ! Process keys for spec file
;    79          0147  1     INPUT:              JSB_INPUT,                  ! General input routine
;    80          0148  1     COMPARE:            JSB_COMPARE,                ! General compare routine
;    81          0149  1     SOR$$COMPATIBLE:    CAL_CTXREG,                 ! Test keys for compatibility
;    82          0150  1     CLEAN_UP:           CAL_CTXREG NOVALUE;         ! Release resources
;    83          0151  1
;    84          0152  1 SOR$$END_ROUTINE_(CLEAN_UP);                        ! Declare a clean-up routine
;    85          0153  1
;    86          0154  1 EXTERNAL ROUTINE
;    87          0155  1     LIB$SFREE1_DD:      ADDRESSING_MODE(GENERAL),       ! Free a dynamic string
;    88          0156  1     LIB$GET_VM:         ADDRESSING_MODE(GENERAL),       ! Get virtual memory
;    89          0157  1     STR$APPEND:         ADDRESSING_MODE(GENERAL),       ! Append strings
;    90          0158  1     SOR$$SFPRS:         CAL_CTXREG,                     ! Parse specifications
;    91          0159  1     SOR$$BEST_FILE_NAME:CAL_CTXREG NOVALUE,             ! Get best file name string
;    92          0160  1     SOR$$ALLOCATE:      CAL_CTXREG,                     ! Allocate storage
;    93          0161  1     SOR$$DEALLOCATE:    CAL_CTXREG NOVALUE,             ! Deallocate storage
;    94          0162  1     SOR$$KEY_SUB:       CAL_CTXREG,                     ! Generate routines
;    95          0163  1     SOR$$ERROR;                                         ! Error routine
;    96          0164  1
;    97          0165  1 ! Define offsets within the internal format record
;    98          0166  1 !
;    99          0167  1 LITERAL
;   100          0168  1     OFF_STAB=   0,      ! Offset to the stable information      (long)
;   101          0169  1     OFF_FMT=    4,      ! Offset to the format number           (byte)
;   102          0170  1     OFF_LEN=    5,      ! Offset to the record length           (word)
;   103          0171  1     OFF_ADR=    7;      ! Offset to the data portion of the record
;   104          0172  1
;   105          0173  1 ! Define offsets for use by SOR$$COMPARE.
;   106          0174  1 !
;   107          0175  1 BIND ZIP_CTX = 0:       BLOCK[CTX_K_SIZE] FIELD(CTX_FIELDS);
;   108          0176  1 GLOBAL LITERAL
;   109          0177  1     COM$L_COLLATE=      ZIP_CTX[COM_COLLATE],
;   110          0178  1     COM$B_PAD=          ZIP_CTX[COM_PAD];
```

```
;  112          0179  1 GLOBAL ROUTINE SOR$$SPEC_FILE: CAL_CTXREG =
;  113          0180  1
;  114          0181  1 !++
;  115          0182  1 !
;  116          0183  1 ! FUNCTIONAL DESCRIPTION:
;  117          0184  1 !
;  118          0185  1 !       This routine processes the specification text.
;  119          0186  1 !
;  120          0187  1 ! FORMAL PARAMETERS:
;  121          0188  1 !
;  122          0189  1 !       NONE
;  123          0190  1 !
;  124          0191  1 ! IMPLICIT INPUTS:
;  125          0192  1 !
;  126          0193  1 !       NONE
;  127          0194  1 !
;  128          0195  1 ! IMPLICIT OUTPUTS:
;  129          0196  1 !
;  130          0197  1 !       NONE
;  131          0198  1 !
;  132          0199  1 ! ROUTINE VALUE:
;  133          0200  1 !
;  134          0201  1 !       Status code.
;  135          0202  1 !
;  136          0203  1 ! SIDE EFFECTS:
;  137          0204  1 !
;  138          0205  1 !       NONE
;  139          0206  1 !
;  140          0207  1 !--
;  141          0208  2     BEGIN
;  142          0209  2     EXTERNAL REGISTER
;  143          0210  2         CTX =   COM_REG_CTX:    REF BLOCK[CTX_K_SIZE]
;  144          0211  2                                 FIELD(CTX_FIELDS);
;  145          0212  2     LOCAL
;  146          0213  2         FAB:    $FAB_DECL,              ! FAB block
;  147          0214  2         NAM:    $NAM_DECL VOLATILE,     ! NAM block
;  148          0215  2         RAB:    REF $RAB_DECL,          ! RAB block
;  149          0216  2         DDB:    REF DDB_BLOCK,
;  150          0217  2         FNA:    BLOCK[NAM$C_MAXRSS, BYTE],      ! File name string area
;  151          0218  2         BUF:    VECTOR[MAX_SPC_LINE,BYTE],      ! Buffer area
;  152          0219  2         DESC:   BLOCK[8,BYTE],                  ! Dynamic string descriptor
;  153          0220  2         STATUS;                 ! Status
;  154          0221  2
;  155          0222  2
;  156          0223  2     ! Initialize the FAB (file access block) and the NAM (name block)
;  157          0224  2     !
;  158        P 0225  2     $FAB_INIT(
;  159        P 0226  2         FAB = FAB[BASE_],                   ! FAB block
;  160        P 0227  2         NAM = NAM[BASE_],                   ! NAM block
;  161        P 0228  2 !       FNA                                 ! File name area        (set below)
;  162        P 0229  2 !       FNS                                 ! File name area size   (set below)
;  163        P 0230  2         FAC = GET,                          ! File access
;  164        P 0231  2         SHR = GET,                          ! Sharing
;  165        P 0232  2         DNA = UPLIT BYTE(STR_SPC_EXT),      ! Default extension is .SRT
;  166        P 0233  2         DNS = %CHARCOUNT(STR_SPC_EXT),      ! Default extension is .SRT
;  167        P 0234  2         RFM = VAR,                          ! Needed if no input files
;  168          0235  2         RAT = CR);                          ! Record attributes
```

```
;  169        P 0236  2      $NAM_INIT(
;  170        P 0237  2          NAM = NAM[BASE_],                   ! NAM block
;  171        P 0238  2          ESS = %ALLOCATION(FNA),             ! Expanded name string size
;  172        P 0239  2          ESA = FNA[BASE_],                   ! Expanded name string area
;  173        P 0240  2          RSS = %ALLOCATION(FNA),             ! Resultant name string size
;  174           0241  2          RSA = FNA[BASE_]);                  ! Resultant name string area
;  175           0242  2
;  176           0243  2
;  177           0244  2      ! Initialize a dynamic string descriptor for the text
;  178           0245  2      !
;  179           0246  2      DESC[DSC$W_LENGTH] = 0;
;  180           0247  2      DESC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  181           0248  2      DESC[DSC$B_CLASS] = DSC$K_CLASS_D;
;  182           0249  2      DESC[DSC$A_POINTER] = 0;
;  183           0250  2
;  184           0251  2
;  185           0252  2      ! Loop for each input file
;  186           0253  2      !
;  187           0254  2      DDB = .CTX[COM_SPC_DDB];                    ! Point to first DDB
;  188           0255  2      WHILE .DDB[BASE_] NEQ 0 DO
;  189           0256  3          BEGIN
;  190           0257  3
;  191           0258  3          ! Actually open the input file
;  192           0259  3          !
;  193           0260  3          NAM[NAM$B_RSL] = 0;
;  194           0261  3          NAM[NAM$B_ESL] = 0;
;  195           0262  3          FAB[FAB$W_IFI] = 0;
;  196           0263  4          BEGIN
;  197           0264  4          SWITCHES STRUCTURE(BLOCK[,BYTE]);
;  198           0265  4          FAB[FAB$B_FNS] = .DDB[DDB_NAME][DSC$W_LENGTH];
;  199           0266  4          FAB[FAB$L_FNA] = .DDB[DDB_NAME][DSC$A_POINTER];
;  200           0267  4          END;
;  201           0268  3          STATUS = $OPEN(FAB = FAB[BASE_]);
;  202           0269  3
;  203           0270  3
;  204           0271  3          ! Get the best file name string available into NAM$B_RSL/NAM$L_RSA
;  205           0272  3          !
;  206           0273  3          SOR$$BEST_FILE_NAME(FAB[BASE_], DDB[DDB_NAME]);
;  207           0274  3
;  208           0275  3          IF NOT .FAB[FAB$L_STS]
;  209           0276  3          THEN
;  210           0277  3              RETURN SOR$$ERROR(SOR$_SHR_OPENIN, 1, DDB[DDB_NAME],
;  211           0278  3                      .FAB[FAB$L_STS], .FAB[FAB$L_STV]);
;  212           0279  3
;  213           0280  3          ! Connect to the FAB
;  214           0281  3          !
;  215           0282  3          RAB = DDB[DDB_RAB+BASE_];
;  216        P  0283  3          $RAB_INIT(
;  217        P  0284  3              RAB = RAB[BASE_],
;  218        P  0285  3              FAB = FAB[BASE_],
;  219        P  0286  3              RAC = SEQ,
;  220        P  0287  3              USZ = %ALLOCATION(BUF),
;  221        P  0288  3              UBF = BUF,
;  222           0289  3              ROP = <RAH,LOC,MAS>);
;  223           0290  3
;  224           0291  3          STATUS = $CONNECT(RAB = RAB[BASE_]);
;  225           0292  3          IF NOT .STATUS
```

```
;   226     0293  3          THEN
;   227     0294  3              RETURN SOR$$ERROR(SOR$_SHR_OPENOUT, 1, DDB[DDB_NAME],
;   228     0295  3                      .RAB[RAB$L_STS], .RAB[RAB$L_STV]);
;   229     0296  3
;   230     0297  3          ! Read all the records from the file
;   231     0298  3          !
;   232     0299  3          WHILE TRUE DO
;   233     0300  4              BEGIN
;   234     0301  4
;   235     0302  5              IF (STATUS = $GET(RAB = RAB[BASE_]))
;   236     0303  4              THEN
;   237     0304  5                  BEGIN
;   238     0305  5                  LOCAL
;   239     0306  5                      D:  VECTOR[2];              ! Descriptor
;   240     0307  5
;   241     0308  5                  ! Append the record and a null to the string
;   242     0309  5                  !
;   243     0310  5                  D[0] = .RAB[RAB$W_RSZ];
;   244     0311  5                  D[1] = .RAB[RAB$L_RBF];
;   245     0312  5                  DECR I FROM 1 TO 0 DO
;   246     0313  6                      BEGIN
;   247     0314  6                      STATUS = STR$APPEND(DESC[BASE_], D[0]);
;   248     0315  6                      IF NOT .STATUS
;   249     0316  6                      THEN
;   250     0317  6                          RETURN SOR$$ERROR(SOR$_SHR_SYSERROR, 0, .STATUS);
;   251     0318  6                      D[0] = 1;
;   252     0319  6                      D[1] = UPLIT BYTE(0);
;   253     0320  5                      END;
;   254     0321  5                  END
;   255     0322  4              ELIF
;   256     0323  4                  .STATUS EQL RMS$_RSA            ! Record Stream Active
;   257     0324  4                  THEN
;   258     0325  5                      $WAIT(RAB=RAB[BASE_])       ! Wait until not so active
;   259     0326  4                  ELSE
;   260     0327  4                      EXITLOOP;                   ! Some other error
;   261     0328  3                  END;
;   262     0329  3
;   263     0330  3
;   264     0331  3          ! Check for the expected status
;   265     0332  3          !
;   266     0333  3          IF .STATUS NEQ RMS$_EOF
;   267     0334  3          THEN
;   268     0335  3              SOR$$ERROR(SOR$_SHR_READERR, 1, DDB[DDB_NAME],
;   269     0336  3                      .RAB[RAB$L_STS], .RAB[RAB$L_STV]);
;   270     0337  3
;   271     0338  3
;   272     0339  3          ! All records have been read from this file, so close it.
;   273     0340  3          ! Zero the IFI in the DDB, so we know that this file is closed
;   274     0341  3          !
;   275     0342  3          IF NOT $CLOSE(FAB=FAB[BASE_])
;   276     0343  3          THEN
;   277     0344  3              SOR$$ERROR(SOR$_SHR_CLOSEIN, 1, DDB[DDB_NAME],
;   278     0345  3                  .FAB[FAB$L_STS], .FAB[FAB$L_STV]);
;   279     0346  3          DDB[DDB_IFI] = 0;
;   280     0347  3
;   281     0348  3
;   282     0349  3          ! Advance to the next file
```

```
;  283     0350  3          !
;  284     0351  3          DDB = .DDB[DDB_NEXT];
;  285     0352  3          END;
;  286     0353  3
;  287     0354  3
;  288     0355  2      ! Append any other text to the buffer
;  289     0356  2      !
;  290     0357  2      STATUS = STR$APPEND(DESC[BASE_], CTX[COM_SPC_TXT]);
;  291     0358  2      IF NOT .STATUS THEN RETURN SOR$$ERROR(SOR$_SHR_SYSERROR, 0, .STATUS);
;  292     0359  2
;  293     0360  2
;  294     0361  2      ! Allocate a work area to hold the tables produced by SOR$$SFPRS
;  295     0362  2      !
;  296     0363  2      IF .CTX[COM_WRK_ADR] EQL 0
;  297     0364  2      THEN
;  298     0365  3          BEGIN
;  299     0366  3          CTX[COM_WRK_SIZ] = WRK_K_ALLOC;
;  300     0367  3          STATUS = LIB$GET_VM(CTX[COM_WRK_SIZ], CTX[COM_WRK_ADR]);
;  301     0368  3          IF NOT .STATUS THEN SOR$$ERROR(SOR$_SHR_SYSERROR, 0, .STATUS);
;  302     0369  3          CTX[COM_WRK_END] = .CTX[COM_WRK_ADR] + .CTX[COM_WRK_SIZ];
;  303     0370  3          END;
;  304     0371  2
;  305     0372  2
;  306     0373  2      ! Call SOR$$SFPRS to build the tables
;  307     0374  2      !
;  308     0375  2      BEGIN
;  309     0376  2      LOCAL D: VECTOR[2];          ! Descriptor
;  310     0377  2      D[0] = .DESC[DSC$W_LENGTH];
;  311     0378  2      D[1] = .DESC[DSC$A_POINTER];
;  312     0379  2      STATUS = SOR$$SFPRS(D[0]);
;  313     0380  2      IF NOT .STATUS
;  314     0381  2      THEN
;  315     0382  3          RETURN SOR$$FATAL(.STATUS);
;  316     0383  2      END;
;  317     0384  2
;  318     0385  2
;  319     0386  2      ! Free the dynamic strings
;  320     0387  2      !
;  321     0388  2      STATUS = LIB$SFREE1_DD(DESC[BASE_]);          ! Free the string
;  322     0389  2      IF NOT .STATUS THEN SOR$$ERROR(SOR$_SHR_SYSERROR, 0, .STATUS);
;  323     0390  2      STATUS = LIB$SFREE1_DD(CTX[COM_SPC_TXT]);     ! Free the string
;  324     0391  2      IF NOT .STATUS THEN SOR$$ERROR(SOR$_SHR_SYSERROR, 0, .STATUS);
;  325     0392  2
;  326     0393  2      RETURN SS$_NORMAL;
;  327     0394  1      END;


                                                        .TITLE  SOR$SPEC_FILE
                                                        .IDENT  \V04-000\

                                                        .PSECT  SOR$RO_CODE_____________________2,NOWRT,  SHR,  PIC,

                                        00000000V 00000 _CLEAN_UP:
                                                        .LONG   <CLEAN_UP-_CLEAN_UP>                            ;

                                                        .PSECT  SOR$RO_CODE,NOWRT,  SHR,  PIC,2
```

```
                                                54  52  53  2E  00000 P.AAA:  .ASCII  \.SRT\                                                ;
                                                            00  00004 P.AAB:  .BYTE   0                                                     ;

                                                                              ZIP_CTX=            0
                                                                              COM$L_COLLATE==     104
                                                                              COM$B_PAD==         257
                                                                                    .EXTRN  LIB$SFREE1_DD, LIB$GET_VM
                                                                                    .EXTRN  STR$APPEND, SOR$$SFPRS
                                                                                    .EXTRN  SOR$$BEST_FILE_NAME
                                                                                    .EXTRN  SOR$$ALLOCATE, SOR$$DEALLOCATE
                                                                                    .EXTRN  SOR$$KEY_SUB, SOR$$ERROR
                                                                                    .EXTRN  SYS$OPEN, SYS$CONNECT
                                                                                    .EXTRN  SYS$GET, SYS$WAIT
                                                                                    .EXTRN  SYS$CLOSE

                                                                  07FC 00000        .ENTRY  SOR$$SPEC_FILE, Save R2,R3,R4,R5,R6,R7,R8,-  ; 0179
                                                                                            R9,R10
                                                        5A 00000000G  00  9E 00002          MOVAB   SOR$$ERROR, R10                         ;
                                                        5E      FDBC  CE  9E 00009          MOVAB   -580(SP), SP                            ;
                                    0050  8F      00        6E        00  2C 0000E          MOVC5   #0, (SP), #0, #80, $RMS_PTR             ; 0235
                                                                B0    AD     00015
                                                    B0  AD      5003  8F  B0 00017          MOVW    #20483, $RMS_PTR                        ;
                                                    C6  AD      0202  8F  B0 0001D          MOVW    #514, $RMS_PTR+22                       ;
                                                    CE  AD      0202  8F  B0 00023          MOVW    #514, $RMS_PTR+30                       ;
                                                    D8  AD      FF50  CD  9E 00029          MOVAB   NAM, $RMS_PTR+40                        ;
                                                    E0  AD        C9  AF  9E 0002F          MOVAB   P.AAA, $RMS_PTR+48                      ;
                                                    E5  AD            04  90 00034          MOVB    #4, $RMS_PTR+53                         ;
                                    0060  8F      00        6E        00  2C 00038          MOVC5   #0, (SP), #0, #96, $RMS_PTR             ; 0241
                                                            FF50      CD     0003F
                                                FF50  CD      6002  8F  B0 00042          MOVW    #24578, $RMS_PTR                        ;
                                                FF52  CD            01  8E 00049          MNEGB   #1, $RMS_PTR+2                          ;
                                                FF54  CD      0094  CE  9E 0004E          MOVAB   FNA, $RMS_PTR+4                         ;
                                                FF5A  CD            01  8E 00055          MNEGB   #1, $RMS_PTR+10                         ;
                                                FF5C  CD      0094  CE  9E 0005A          MOVAB   FNA, $RMS_PTR+12                        ;
                                                  08  AE  020E0000  8F  D0 00061          MOVL    #34471936, DESC                         ; 0246
                                                            0C  AE  D4 00069            CLRL    DESC+4                                  ; 0249
                                                      57    00AC  CB  D0 0006C          MOVL    172(CTX), DDB                           ; 0254
                                                                03  12 00071 1$:         BNEQ    2$                                      ; 0255
                                                              0121  31 00073            BRW     11$                                     ;
                                                          FF53  CD  94 00076 2$:         CLRB    NAM+3                                   ; 0260
                                                          FF5B  CD  94 0007A            CLRB    NAM+11                                  ; 0261
                                                            B2  AD  B4 0007E            CLRW    FAB+2                                   ; 0262
                                                      58    04  A7  9E 00081          MOVAB   4(DDB), R8                              ; 0265
                                                    E4  AD      68  90 00085          MOVB    (R8), FAB+52                            ;
                                                    DC  AD    04  A8  D0 00089          MOVL    4(R8), FAB+44                           ; 0266
                                                        B0    AD  9F 0008E            PUSHAB  FAB                                     ; 0268
                                            00000000G   00        01  FB 00091          CALLS   #1, SYS$OPEN                            ;
                                                      59          50  D0 00098          MOVL    R0, STATUS                              ;
                                                                  58  DD 0009B          PUSHL   R8                                      ; 0273
                                                        B0        AD  9F 0009D          PUSHAB  FAB                                     ;
                                            00000000G   00        02  FB 000A0          CALLS   #2, SOR$$BEST_FILE_NAME                 ;
                                                      10    B8    AD  E8 000A7          BLBS    FAB+8, 3$                               ; 0275
                                                      7E    B8    AD  7D 000AB          MOVQ    FAB+8, -(SP)                            ; 0278
                                                                  58  DD 000AF          PUSHL   R8                                      ; 0277
                                                                  01  DD 000B1          PUSHL   #1                                      ;
                                                        001C109C  8F  DD 000B3          PUSHL   #1839260                                ;
                                                                  48  11 000B9          BRB     4$                                      ;
```

```
                        56        14   A7  9E 000BB 3$:     MOVAB   20(R7), RAB                             ; 0282
0044   8F         00      6E             00  2C 000BF       MOVC5   #0, (SP), #0, #68, (RAB)                ; 0289
                                         66      000C6
                        66      4401     8F  B0 000C7       MOVW    #17409, (RAB)                           ;
                    04  A6  00010220     8F  D0 000CC       MOVL    #66080, 4(RAB)                          ;
                            1E           A6  94 000D4       CLRB    30(RAB)                                 ;
                    20  A6        84     8F  9B 000D7       MOVZBW  #132, 32(RAB)                           ;
                    24  A6        10     AE  9E 000DC       MOVAB   BUF, 36(RAB)                            ;
                    3C  A6        B0     AD  9E 000E1       MOVAB   FAB, 60(RAB)                            ;
                                         56  DD 000E6       PUSHL   RAB                                     ; 0291
                00000000G   00           01  FB 000E8       CALLS   #1, SYS$CONNECT                         ;
                            59           50  D0 000EF       MOVL    R0, STATUS                              ;
                            12           59  E8 000F2       BLBS    STATUS, 5$                              ; 0292
                            7E        08 A6  7D 000F5       MOVQ    8(RAB), -(SP)                           ; 0295
                                         58  DD 000F9       PUSHL   R8                                      ; 0294
                                         01  DD 000FB       PUSHL   #1                                      ;
                            001C10A4     8F  DD 000FD       PUSHL   #1839268                                ;
                            6A           05  FB 00103 4$:   CALLS   #5, SOR$$ERROR                         ;
                                         04 00106           RET                                             ;
                                         56  DD 00107 5$:   PUSHL   RAB                                     ; 0302
                00000000G   00           01  FB 00109       CALLS   #1, SYS$GET                             ;
                            59           50  D0 00110       MOVL    R0, STATUS                              ;
                            2C           59  E9 00113       BLBC    STATUS, 7$                              ;
                            6E        22 A6  3C 00116       MOVZWL  34(RAB), D                              ; 0310
                    04      AE        28 A6  D0 0011A       MOVL    40(RAB), D+4                            ; 0311
                            52           01  D0 0011F       MOVL    #1, I                                   ; 0312
                                         5E  DD 00122 6$:   PUSHL   SP                                      ; 0314
                                      0C AE  9F 00124       PUSHAB  DESC                                    ;
                00000000G   00           02  FB 00127       CALLS   #2, STR$APPEND                          ;
                            59           50  D0 0012E       MOVL    R0, STATUS                              ;
                            77           59  E9 00131       BLBC    STATUS, 12$                             ; 0315
                            6E           01  D0 00134       MOVL    #1, D                                   ; 0318
                    04      AE      FEC4 CF  9E 00137       MOVAB   P.AAB, D+4                              ; 0319
                            E2           52  F4 0013D       SOBGEQ  I, 6$                                   ; 0312
                                         C5  11 00140       BRB     5$                                      ; 0302
            000182DA    8F               59  D1 00142 7$:   CMPL    STATUS, #99034                          ; 0323
                                         0B  12 00149       BNEQ    8$                                      ;
                                         56  DD 0014B       PUSHL   RAB                                     ; 0325
                00000000G   00           01  FB 0014D       CALLS   #1, SYS$WAIT                            ;
                                         B1  11 00154       BRB     5$                                      ;
            0001827A    8F               59  D1 00156 8$:   CMPL    STATUS, #98938                          ; 0333
                                         11  13 0015D       BEQL    9$                                      ;
                            7E        08 A6  7D 0015F       MOVQ    8(RAB), -(SP)                           ; 0336
                                         58  DD 00163       PUSHL   R8                                      ; 0335
                                         01  DD 00165       PUSHL   #1                                      ;
                            001C10B2     8F  DD 00167       PUSHL   #1839282                                ;
                            6A           05  FB 0016D       CALLS   #5, SOR$$ERROR                          ;
                                      B0 AD  9F 00170 9$:   PUSHAB  FAB                                     ; 0342
                00000000G   00           01  FB 00173       CALLS   #1, SYS$CLOSE                           ;
                            11           50  E8 0017A       BLBS    R0, 10$                                 ;
                            7E        B8 AD  7D 0017D       MOVQ    FAB+8, -(SP)                            ; 0345
                                         58  DD 00181       PUSHL   R8                                      ; 0344
                                         01  DD 00183       PUSHL   #1                                      ;
                            001C1052     8F  DD 00185       PUSHL   #1839186                                ;
                            6A           05  FB 0018B       CALLS   #5, SOR$$ERROR                          ;
                                      0C A7  D4 0018E 10$:   CLRL    12(DDB)                                 ; 0346
                            57           67  D0 00191       MOVL    (DDB), DDB                              ; 0351
```

```
                                FEDA   31  00194          BRW     1$                                      ; 0255
                          00F4   CB   9F  00197 11$:     PUSHAB  244(CTX)                                ; 0357
                            0C   AE   9F  0019B          PUSHAB  DESC
                00000000G   00        02  0019E          CALLS   #2, STR$APPEND
                            59        50  001A5          MOVL    R0, STATUS
                            0E        59  001A8          BLBS    STATUS, 13$                             ; 0358
                                      59  001AB 12$:     PUSHL   STATUS
                                      7E  D4  001AD          CLRL    -(SP)
                        001C11B4      8F  DD  001AF          PUSHL   #1839540
                            6A        03  FB  001B5          CALLS   #3, SOR$$ERROR
                                      04  001B8          RET
                            53  0128  CB  9E  001B9 13$:     MOVAB   296(CTX), R3                    ; 0363
                                      63  D5  001BE          TSTL    (R3)
                                      2E  12  001C0          BNEQ    15$
                            52  0124  CB  9E  001C2          MOVAB   292(CTX), R2                    ; 0366
                            62 00010000   8F  D0  001C7          MOVL    #65536, (R2)
                                      0C  BB  001CE          PUSHR   #^M<R2,R3>                      ; 0367
                00000000G   00        02  FB  001D0          CALLS   #2, LIB$GET_VM
                            59        50  D0  001D7          MOVL    R0, STATUS
                            0D        59  E8  001DA          BLBS    STATUS, 14$                     ; 0368
                                      59  DD  001DD          PUSHL   STATUS
                                      7E  D4  001DF          CLRL    -(SP)
                        001C11B4      8F  DD  001E1          PUSHL   #1839540
                            6A        03  FB  001E7          CALLS   #3, SOR$$ERROR
        012C    CB          63        62  C1  001EA 14$:     ADDL3   (R2), (R3), 300(CTX)            ; 0369
                            6E    08  AE  3C  001F0 15$:     MOVZWL  DESC, D                         ; 0377
                        04  AE    0C  AE  D0  001F4          MOVL    DESC+4, D+4                     ; 0378
                                      5E  DD  001F9          PUSHL   SP                              ; 0379
                00000000G   00        01  FB  001FB          CALLS   #1, SOR$$SFPRS
                            59        50  D0  00202          MOVL    R0, STATUS
                            0C        59  E8  00205          BLBS    STATUS, 16$                     ; 0380
                    50      59        07  CB  00208          BICL3   #7, STATUS, R0                  ; 0382
                    7E      50        04  C9  0020C          BISL3   #4, R0, -(SP)
                            6A        01  FB  00210          CALLS   #1, SOR$$ERROR
                                      04  00213          RET
                          08  AE   9F  00214 16$:     PUSHAB  DESC                                    ; 0388
                00000000G   00        01  FB  00217          CALLS   #1, LIB$SFREE1_DD
                            59        50  D0  0021E          MOVL    R0, STATUS
                            0D        59  E8  00221          BLBS    STATUS, 17$                     ; 0389
                                      59  DD  00224          PUSHL   STATUS
                                      7E  D4  00226          CLRL    -(SP)
                        001C11B4      8F  DD  00228          PUSHL   #1839540
                            6A        03  FB  0022E          CALLS   #3, SOR$$ERROR
                          00F4   CB   9F  00231 17$:     PUSHAB  244(CTX)                                ; 0390
                00000000G   00        01  FB  00235          CALLS   #1, LIB$SFREE1_DD
                            59        50  D0  0023C          MOVL    R0, STATUS
                            0D        59  E8  0023F          BLBS    STATUS, 18$                     ; 0391
                                      59  DD  00242          PUSHL   STATUS
                                      7E  D4  00244          CLRL    -(SP)
                        001C11B4      8F  DD  00246          PUSHL   #1839540
                            6A        03  FB  0024C          CALLS   #3, SOR$$ERROR
                            50        01  D0  0024F 18$:     MOVL    #1, R0                          ; 0393
                                      04  00252          RET                                         ; 0394

; Routine Size:  595 bytes,    Routine Base:  SOR$RO_CODE + 0005
```

```
;  329     0395  1 ROUTINE CALC_LRL_OUT: CAL_CTXREG NOVALUE =
;  330     0396  1
;  331     0397  1 !++
;  332     0398  1 !
;  333     0399  1 ! FUNCTIONAL DESCRIPTION:
;  334     0400  1 !
;  335     0401  1 !       Do some processing of the spec tables after the input LRL is known.
;  336     0402  1 !       It determines the longest output record length.
;  337     0403  1 !
;  338     0404  1 ! FORMAL PARAMETERS:
;  339     0405  1 !
;  340     0406  1 !       NONE
;  341     0407  1 !
;  342     0408  1 ! IMPLICIT INPUTS:
;  343     0409  1 !
;  344     0410  1 !       NONE
;  345     0411  1 !
;  346     0412  1 ! IMPLICIT OUTPUTS:
;  347     0413  1 !
;  348     0414  1 !       CTX[COM_LRL_OUT]        Longest output record length
;  349     0415  1 !       CTX[COM_SPEC_TKS]       Total key size
;  350     0416  1 !       CTX[COM_FORMATS]        Number of different record formats
;  351     0417  1 !       CTX[COM_VAR]            Flag indicating variable-length records
;  352     0418  1 !       FDT[0,FDT_FLD_SIZ]      Input LRL
;  353     0419  1 !       KFT[*,KFT_NDE_SIZ]      Input LRL (only those that refer to first FDT)
;  354     0420  1 !       KFT[*,KFT_NDE_POS]      Position of field in internal node
;  355     0421  1 !       KFT[*,KFT_BUI[D]        True if field must be built/copied
;  356     0422  1 !
;  357     0423  1 ! ROUTINE VALUE:
;  358     0424  1 !
;  359     0425  1 !       Status code.
;  360     0426  1 !
;  361     0427  1 ! SIDE EFFECTS:
;  362     0428  1 !
;  363     0429  1 !       NONE
;  364     0430  1 !
;  365     0431  1 !--
;  366     0432  2     BEGIN
;  367     0433  2     EXTERNAL REGISTER
;  368     0434  2         CTX =   COM_REG_CTX:    REF BLOCK[CTX_K_SIZE]
;  369     0435  2                                 FIELD(CTX_FIELDS);
;  370     0436  2     BIND
;  371     0437  2         RDT = CTX[COM_RDT_ADR]: REF RDT_TAB[],  ! Record definition table
;  372     0438  2         KFT = CTX[COM_KFT_ADR]: REF KFT_TAB[],  ! Key field table
;  373     0439  2         FDT = CTX[COM_FDT_ADR]: REF FDT_TAB[],  ! Field definition table
;  374     0440  2         CFT = CTX[COM_CFT_ADR]: REF CFT_TAB[];  ! Constant definition table
;  375     0441  2
;  376     0442  2     LOCAL
;  377     0443  2         SEEN: BITVECTOR[KFT_MAX],
;  378     0444  2         MAX_DSUM,
;  379     0445  2         MAX_KSUM;
;  380     0446  2
;  381     0447  2
;  382     0448  2     ! Store the input LRL in:
;  383     0449  2     !   FDT[0,FDT_FLD_SIZ] and KFT[*,KFT_NDE_SIZ] for every KFT entry
;  384     0450  2     !   with KFT_CONSTANT = FALSE and KFT_FDT_IDX = 0.
;  385     0451  2     !
```

```
;  386          0452  3      BEGIN
;  387          0453  3      LOCAL
;  388          0454  3          KFT_PTR: REF KFT_TAB[];          ! Local pointer to KFT table
;  389          0455  3      FDT[0,FDT_FLD_SIZ] = .CTX[COM_LRL];
;  390          0456  3      KFT_PTR = KFT[0,BASE_];
;  391          0457  3      DECR I FROM .CTX[COM_KFT_SIZ]-1 TO 0 DO
;  392          0458  4          BEGIN
;  393          0459  4          IF NOT .KFT_PTR[0,KFT_CONSTANT] AND .KFT_PTR[0,KFT_FDT_IDX] EQL 0
;  394          0460  4          THEN
;  395          0461  4              KFT_PTR[0,KFT_NDE_SIZ] = .CTX[COM_LRL];
;  396          0462  4          KFT_PTR = KFT_PTR[1,BASE_];
;  397          0463  4          END;
;  398          0464  3      END;
;  399          0465  3
;  400          0466  3      ! Initialize our variables
;  401          0467  3      !
;  402          0468  3      CH$FILL(0, %ALLOCATION(SEEN), SEEN[0]);
;  403          0469  3      MAX_DSUM = 0;
;  404          0470  3      MAX_KSUM = 0;
;  405          0471  3
;  406          0472  3      ! Loop through all record definitions for include statements
;  407          0473  3      !
;  408          0474  3      DECR RDT_IX FROM .CTX[COM_RDT_SIZ]-1 TO 0 DO
;  409          0475  3      IF .RDT[.RDT_IX, RDT_INCLUDE]
;  410          0476  3      THEN
;  411          0477  3          BEGIN
;  412          0478  3          BUILTIN
;  413          0479  3              TESTBITSS;
;  414          0480  3          LOCAL
;  415          0481  3              Z;
;  416          0482  3
;  417          0483  3          ! Have we seen this before?
;  418          0484  3          !
;  419          0485  3          Z = .RDT[.RDT_IX, RDT_KFT_IDX];
;  420          0486  3          IF TESTBITSS(SEEN[.Z])
;  421          0487  3          THEN
;  422          0488  4              BEGIN
;  423        C 0489  4 %(              ! Find the RDT entry, and copy relevant information
;  424        C 0490  4                 !
;  425        C 0491  4                 DECR TMP_IX FROM .CTX[COM_RDT_SIZ]-1 TO 0 DO
;  426        C 0492  4                 IF .RDT[.TMP_IX, RDT_INCLUDE]
;  427        C 0493  4                 THEN
;  428        C 0494  4                     BEGIN
;  429        C 0495  4                     IF .Z EQL .RDT[.TMP_IX, RDT_KFT_IDX]
;  430        C 0496  4                     THEN
;  431        C 0497  4                         BEGIN
;  432        C 0498  4                         ! currently there's no relevant info to copy
;  433        C 0499  4                         EXITLOOP;
;  434        C 0500  4                         END;
;  435        C 0501  4                     END;
;  436          0502  4 )%                0;
;  437          0503  4                   END
;  438          0504  3          ELSE
```

```
;  443          0509  4          BEGIN
;  444          0510  4          LOCAL
;  445          0511  4              DSUM,                           ! Sum of data lengths
;  446          0512  4              KSUM,                           ! Sum of key lengths
;  447          0513  4              KFT_PTR: REF KFT_TAB[];         ! Local pointer to KFT table
;  448          0514  4
;  449          0515  4          ! Increment the number of different record formats
;  450          0516  4          !
;  451          0517  4          CTX[COM_FORMATS] = .CTX[COM_FORMATS] + 1;
;  452          0518  4
;  453          0519  4          KFT_PTR = KFT[.Z,BASE_];            ! Pointer to key field entry
;  454          0520  4          DSUM = 0;
;  455          0521  4          KSUM = 0;
;  456          0522  5          IF ONEOF_(.CTX[COM_SORT_TYPE], BMSK_(TYP_K_ADDRESS,TYP_K_INDEX))
;  457          0523  4          THEN
;  458          0524  4              DSUM = RAB$S_RFA;
;  459          0525  4          WHILE 1 DO
;  460          0526  5              BEGIN
;  461          0527  5              LOCAL L;
;  462          0528  5              L = .KFT_PTR[0, KFT_NDE_SIZ];   ! Get length in bytes
;  463          0529  5              IF .KFT_PTR[0, KFT_DATA]        ! Data or key?
;  464          0530  5              THEN
;  465        P 0531  6                  BEGIN
;  466        P 0532  6                  IF NOT ONEOF (.CTX[COM_SORT_TYPE],
;  467          0533  7                      BMSK_(TYP_K_ADDRESS,TYP_K_INDEX))
;  468          0534  6                  THEN
;  469          0535  7                      BEGIN
;  470          0536  7                      KFT_PTR[0, KFT_NDE_POS] = .DSUM;
;  471          0537  7                      DSUM = .DSUM + .L;
;  472          0538  7                      END
;  473          0539  6                  ELSE
;  474          0540  6                      KFT_PTR[0, KFT_BUILD] = FALSE;
;  475          0541  6                  END
;  476          0542  5              ELSE
;  477          0543  6                  BEGIN
;  478        P 0544  6                  IF NOT ONEOF (.CTX[COM_SORT_TYPE],
;  479          0545  7                      BMSK_(TYP_K_ADDRESS,TYP_K_INDEX))
;  480          0546  6                  THEN
;  481          0547  7                      BEGIN
;  482          0548  7                      KFT_PTR[0, KFT_NDE_POS] = .KSUM;
;  483          0549  7                      KSUM = .KSUM + .L;
;  484          0550  7                      END
;  485          0551  6                  ELSE
;  486          0552  7                      BEGIN
;  487          0553  7                      KFT_PTR[0, KFT_NDE_POS] = .DSUM;
;  488          0554  7                      DSUM = .DSUM + .L;
;  489          0555  6                      END;
;  490          0556  5                  END;
;  491          0557  5              WHILE .KFT_PTR[0,KFT_CONDX] DO  ! ??? Were these ever verified?
;  492          0558  6                  BEGIN
;  493          0559  6                  KFT_PTR = KFT_PTR[1,BASE_];
;  494          0560  6                  KFT_PTR[0, KFT_NDE_POS] = .KFT_PTR[-1, KFT_NDE_POS];
;  495          0561  6                  IF NOT .KFT_PTR[-1, KFT_BUILD]
;  496          0562  6                  THEN
;  497          0563  6                      KFT_PTR[0, KFT_BUILD] = FALSE;
;  498          0564  5                  END;
;  499          0565  5              IF NOT .KFT_PTR[0,KFT_CONTINUE] THEN EXITLOOP;
```

```
5              KFT_PTR = KFT_PTR[1,BASE_];
4              END;
4
4          ! Store the information for this RDT entry
4          !
4          !RDT[.RDT_IX, field] = value;       ! Currently, nothing to store
4
4          ! Update MAX_KSUM
4          !
4          IF .KSUM GTR .MAX_KSUM THEN MAX_KSUM = .KSUM;
4
4          ! Update MAX_DSUM
4          !
4          IF .MAX_DSUM EQL 0
4          THEN
4              MAX_DSUM = .DSUM
4          ELIF .DSUM LSS .MAX_DSUM
4          THEN
4              CTX[COM_VAR] = TRUE
4          ELIF .DSUM GTR .MAX_DSUM
4          THEN
5              BEGIN
5              MAX_DSUM = .DSUM;
5              CTX[COM_VAR] = TRUE;    ! Depends on sort process
5              END
3          END;
2      END;
2
2  ! Store the longest output record length, and total key size
2  !
2  IF .CTX[COM_RDT_SIZ] GTR 0
2  THEN
3      BEGIN
3      CTX[COM_LRL_OUT]  = .MAX_DSUM;  ! Longest output record length
3      CTX[COM_SPEC_TKS] = .MAX_KSUM;  ! Total key size
2      END;
2
1  END;



                                03FC 00000 CALC_LRL_OUT:
                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9        ; 0395
                5E           20   C2 00002         SUBL2   #32, SP
                50     0110  CB   D0 00005         MOVL    272(CTX), R0                ; 0455
           04   A0     0084  CB   B0 0000A         MOVW    132(CTX), 4(R0)
                50     0108  CB   D0 00010         MOVL    264(CTX), KFT_PTR           ; 0456
                58     00FC  CB   9E 00015         MOVAB   252(CTX), R8                ; 0457
                51       01  A8   9A 0001A         MOVZBL  1(R8), I
                         13   11 0001E             BRB     3$
  0B       03   A0           01   E0 00020 1$:     BBS     #1, 3(KFT_PTR), 2$          ; 0459
                         04  A0   95 00025         TSTB    4(KFT_PTR)
                         06   12 00028             BNEQ    2$
           06   A0     0084  CB   B0 0002A         MOVW    132(CTX), 6(KFT_PTR)        ; 0461
                50           08   C0 00030 2$:     ADDL2   #8, KFT_PTR                 ; 0462
```

```
                        EA            51  F4 00033 3$:     SOBGEQ  I, 1$                                   ;0457
              20 00     6E            00  2C 00036         MOVC5   #0, (SP), #0, #32, SEEN                 ;0469
                        6E                00038
                                      57  D4 0003C         CLRL    MAX_DSUM                                ;0470
                                      59  D4 0003E         CLRL    MAX_KSUM                                ;0471
                        50            68  9A 00040         MOVZBL  (R8), RDT_IX                            ;0476
                                  00A1  31 00043 4$:       BRW     16$
                     51 50            06  C5 00046 5$:     MULL3   #6, RDT_IX, R1                          ;0477
                        51      0104  CB  C0 0004A         ADDL2   260(CTX), R1
                        F1            61  E9 0004F         BLBC    (R1), 4$
                        51        04  A1  9A 00052         MOVZBL  4(R1), Z                                ;0487
                     E9 6E            51  E2 00056         BBSS    Z, SEEN, 4$                             ;0488
                        55      0080  CB  9E 0005A         MOVAB   128(CTX), R5                            ;0517
                                  03  A5  96 0005F         INCB    3(R5)
                        51      0108 DB41 7E 00062         MOVAQ   @264(CTX)[Z], KFT_PTR                   ;0519
                                      53  D4 00068         CLRL    DSUM                                    ;0520
                                      56  D4 0006A         CLRL    KSUM                                    ;0521
                  52 18000000   8F    58  AB 78 0006C         ASHL    88(CTX), #402653184, R2                 ;0522
                                      03  18 00075         BGEQ    6$
                        53            06  D0 00077         MOVL    #6, DSUM                                ;0524
                        52        06  A1  3C 0007A 6$:     MOVZWL  6(KFT_PTR), L                           ;0528
                  0D       03     A1  06  E1 0007E         BBC     #6, 3(KFT_PTR), 7$                      ;0529
                  54 18000000   8F    58  AB 78 00083         ASHL    88(CTX), #402653184, R4                 ;0533
                                      15  18 0008C         BGEQ    8$
                                      2A  11 0008E         BRB     10$                                     ;0540
                  54 18000000   8F    58  AB 78 00090 7$:     ASHL    88(CTX), #402653184, R4                 ;0545
                                      08  19 00099         BLSS    8$
                        61            56  B0 0009B         MOVW    KSUM, (KFT_PTR)                         ;0548
                        56            52  C0 0009E         ADDL2   L, KSUM                                 ;0549
                                      06  11 000A1         BRB     9$                                      ;0544
                        61            53  B0 000A3 8$:     MOVW    DSUM, (KFT_PTR)                         ;0553
                        53            52  C0 000A6         ADDL2   L, DSUM                                 ;0554
                  12    03      A1      03  E1 000A9 9$:     BBC     #3, 3(KFT_PTR), 11$                     ;0557
                        51            08  C0 000AE         ADDL2   #8, KFT_PTR                             ;0559
                        61        F8  A1  B0 000B1         MOVW    -8(KFT_PTR), (KFT_PTR)                  ;0560
                  EF    FB      A1      04  E0 000B5         BBS     #4, -5(KFT_PTR), 9$                     ;0561
                        03      A1      10  8A 000BA 10$:    BICB2   #16, 3(KFT_PTR)                         ;0563
                                      E9  11 000BE         BRB     9$                                      ;0557
                        05        03  A1  E9 000C0 11$:    BLBC    3(KFT_PTR), 12$                         ;0565
                        51            08  C0 000C4         ADDL2   #8, KFT_PTR                             ;0566
                                      B1  11 000C7         BRB     6$                                      ;0525
                        59            56  D1 000C9 12$:    CMPL    KSUM, MAX_KSUM                          ;0575
                                      03  15 000CC         BLEQ    13$
                        59            56  D0 000CE         MOVL    KSUM, MAX_KSUM
                                      57  D5 000D1 13$:    TSTL    MAX_DSUM                                ;0579
                                      05  12 000D3         BNEQ    14$
                        57            53  D0 000D5         MOVL    DSUM, MAX_DSUM                          ;0581
                                      0D  11 000D8         BRB     16$
                        57            53  D1 000DA 14$:    CMPL    DSUM, MAX_DSUM                          ;0582
                                      05  19 000DD         BLSS    15$
                                      06  15 000DF         BLEQ    16$                                     ;0585
                        57            53  D0 000E1         MOVL    DSUM, MAX_DSUM                          ;0588
                        65            02  88 000E4 15$:    BISB2   #2, (R5)                                ;0589
                        02            50  F4 000E7 16$:    SOBGEQ  RDT_IX, 17$                             ;0477
                                      03  11 000EA         BRB     18$
                                  FF57  31 000EC 17$:      BRW     5$
                                      68  95 000EF 18$:     TSTB    (R8)                                    ;0596
```

```
                          09  13 000F1          BEQL    19$                                     ;
                  008A    CB 57  B0 000F3          MOVW    MAX_DSUM, 138(CTX)                      ; 0599
                    5E    AB 59  B0 000F8          MOVW    MAX_KSUM, 94(CTX)                       ; 0600
                             04  000FC 19$:       RET                                             ; 0603

; Routine Size:  253 bytes,    Routine Base:  SOR$RO_CODE + 0258
```

```
;  539     0604  1 GLOBAL ROUTINE SOR$$SPEC_KEY_SUB: CAL_CTXREG =
;  540     0605  1
;  541     0606  1 !++
;  542     0607  1 !
;  543     0608  1 ! FUNCTIONAL DESCRIPTION:
;  544     0609  1 !
;  545     0610  1 !       Process key descriptions from the specification text.
;  546     0611  1 !
;  547     0612  1 ! FORMAL PARAMETERS:
;  548     0613  1 !
;  549     0614  1 !       NONE
;  550     0615  1 !
;  551     0616  1 ! IMPLICIT INPUTS:
;  552     0617  1 !
;  553     0618  1 !       CTX             Longword pointing to work area (passed in COM_REG_CTX)
;  554     0619  1 !
;  555     0620  1 !       The following fields of the context area are used as input:
;  556     0621  1 !               COM_SORT_TYPE   Type of sort (TYP_K_RECORD, etc)
;  557     0622  1 !               COM_NUM_FILES   Number of input files
;  558     0623  1 !               COM_LRL         Longest input record length (see below)
;  559     0624  1 !               COM_MINVFC      Length of VFC area
;  560     0625  1 !               COM_COLLATE     Collating sequence information
;  561     0626  1 !               COM_STABLE      Flag indicating stable sort
;  562     0627  1 !               COM_VAR         Flag indicating variable-length records
;  563     0628  1 !               COM_NO_DUPS     Flag indicating to delete duplicate records
;  564     0629  1 !
;  565     0630  1 !       The following fields are used as input/output:
;  566     0631  1 !               COM_COMPARE     Comparison routine
;  567     0632  1 !               COM_EQUAL       Equal-key routine
;  568     0633  1 !               COM_TKS         Total key size (hack hack)
;  569     0634  1 !               COM_SPEC_TKS    Total key size, due to record reformatting
;  570     0635  1 !
;  571     0636  1 !       The following fields are used as output:
;  572     0637  1 !               COM_INPUT       Routine to do input conversion of records
;  573     0638  1 !               COM_LENADR      Routine to return length/address of record
;  574     0639  1 !               COM_SRL         Shortest allowable input record length
;  575     0640  1 !               COM_LRL_INT     Length of internal format record
;  576     0641  1 !
;  577     0642  1 ! IMPLICIT OUTPUTS:
;  578     0643  1 !
;  579     0644  1 !       NONE
;  580     0645  1 !
;  581     0646  1 ! ROUTINE VALUE:
;  582     0647  1 !
;  583     0648  1 !       Status code.
;  584     0649  1 !
;  585     0650  1 ! SIDE EFFECTS:
;  586     0651  1 !
;  587     0652  1 !       NONE
;  588     0653  1 !
;  589     0654  1 !--
;  590     0655  2     BEGIN
;  591     0656  2     EXTERNAL REGISTER
;  592     0657  2         CTX =   COM_REG_CTX:    REF BLOCK[CTX_K_SIZE]
;  593     0658  2                                 FIELD(CTX_FIELDS);
;  594     0659  2     BIND
;  595     0660  2         RDT = CTX[COM_RDT_ADR]: REF RDT_TAB[],  ! Record definition table
```

```
;  596          0661  2          KFT = CTX[COM_KFT_ADR]: REF KFT_TAB[],  ! Key field table
;  597          0662  2          FDT = CTX[COM_FDT_ADR]: REF FDT_TAB[],  ! Field definition table
;  598          0663  2          CFT = CTX[COM_CFT_ADR]: REF CFT_TAB[];  ! Constant definition table
;  599          0664  2
;  600          0665  2      BIND
;  601          0666  2          UE1 = PLIT BYTE(
;  602          0667  2              OPC_PUSHL, M_BD+COM_REG_CTX, %FIELDEXPAND(COM_CTXADR)*%UPVAL,
;  603          0668  2              OPC_PUSHAB, M_BD+COM_REG_SRC2, OFF_LEN,
;  604          0669  2              OPC_PUSHAB, M_BD+COM_REG_SRC1, OFF_LEN,
;  605          0670  2              OPC_PUSHAB, M_BD+COM_REG_SRC2, OFF_ADR,
;  606          0671  2              OPC_PUSHAB, M_BD+COM_REG_SRC1, OFF_ADR,
;  607          0672  2              OPC_CALLS, 5, M_AID+R_PC): VECTOR,
;  608          0673  2          UE2 = PLIT BYTE(
;  609          0674  2              OPC_BLBC, M_R+R_0, 1,
;  610          0675  2              OPC_RSB,
;  611          0676  2              OPC_PUSHL, M_R+R_0,
;  612          0677  2              OPC_PUSHL, 0,
;  613          0678  2              OPC_PUSHL, M_AI+R_PC, LONG(SOR$_RTNERROR),
;  614          0679  2              OPC_CALLS, 3, M_AID+R_PC): VECTOR,
;  615          0680  2          UE3 = PLIT BYTE(
;  616          0681  2              OPC_MOVL, SS$_NORMAL, M_R+R_0,
;  617          0682  2              OPC_RSB): VECTOR,
;  618          0683  2          UE4 = PLIT BYTE(
;  619          0684  2              OPC_BLBC, M_R+R_0, 1,
;  620          0685  2              OPC_RSB,
;  621          0686  2              OPC_MOVL, 1, M_R+R_0,
;  622          0687  2              OPC_CMPL,M_BD+COM_REG_SRC1,OFF_STAB,M_BD+COM_REG_SRC2,OFF_STAB,
;  623          0688  2              OPC_BGTRU, 3,
;  624          0689  2              OPC_SBWC, 1, M_R+R_0,
;  625          0690  2              OPC_RSB): VECTOR;
;  626          0691  2
;  627          0692  2      ROUTINE APPEND(LEN, ADR): CAL_CTXREG NOVALUE =
;  628          0693  2          BEGIN
;  629          0694  2          EXTERNAL REGISTER
;  630          0695  2              CTX = COM_REG_CTX:  REF BLOCK[CTX_K_SIZE]
;  631          0696  2                                  FIELD(CTX_FIELDS);
;  632          0697  2          BIND
;  633          0698  2              XCODE =     CTX[COM_ROUTINES]: VECTOR[2];
;  634          0699  2          LOCAL
;  635          0700  2              DELTA:      VECTOR[2];
;  636          0701  2          DELTA[0] = .XCODE[0] + .LEN;
;  637          0702  2          DELTA[1] = SOR$$ALLOCATE(.DELTA[0]);
;  638          0703  2          CH$MOVE(.LEN, .ADR, CH$MOVE(.XCODE[0], .XCODE[1], .DELTA[1]));
;  639          0704  2          SOR$$DEALLOCATE(.XCODE[0], XCODE[1]);
;  640          0705  2          XCODE[0] = .DELTA[0];
;  641          0706  2          XCODE[1] = .DELTA[1];
;  642          0707  2          END;


                                                00000006  00355              .LONG   6
07 AA 9F 05 A9 9F 05 AA 9F 00 20 00 15 AB DD    00359 P.AAC:     .BYTE   -35, -85, 21, 0, 32, 0, -97, -86, 5, -97, -
                           9F 05 FB 07 A9 9F    00368                    -87, 5, -97, -86, 7, -97, -87, 7, -5, 5, -
                                                                         -97
                                                0036E            .BLKB   1
                                                0036F            .BLKB   2
                                                00000005  00371  .LONG   5
```

```
                8F  DD  00  DD  50  DD  05  01  50  E9  00375 P.AAD:  .BYTE   -23, 80, 1, 5, -35, 80, -35, 0, -35, -113      ;
                                        001C812A    00037F          .LONG   1868074                                        ;
                                    9F  03  FB      00383          .BYTE   -5, 3, -97                                     ;
                                            01      00386          .BLKB   1                                              ;
                                            02      00387          .BLKB   2                                              ;
                                    00000001        00389          .LONG   1                                              ;
                                        05  50  01  DD  0038D P.AAE:  .BYTE   -48, 1, 80, 5                               ;
                                    00000005        00391          .LONG   5                                              ;
D9  03  1A  00  AA  00  A9  D1  50  01  DD  05  01  50  E9  00395 P.AAF:  .BYTE   -23, 80, 1, 5, -48, 1, 80, -47, -87, 0, -  ;
                                        05  50  01      003A4                  -86, 0, 26, 3, -39, 1, 80, 5                   ;
                                                        003A7          .BLKB   2                                              ;

                                                                UE1=            P.AAC
                                                                UE2=            P.AAD
                                                                UE3=            P.AAE
                                                                UE4=            P.AAF


                                                    007C 00000 APPEND:  .WORD   Save R2,R3,R4,R5,R6                        ; 0692
                                        5E              04  C2 00002        SUBL2   #4, SP                                 ;
                                        56          18  AB  9E 00005        MOVAB   24(CTX), R6                            ; 0698
                            7E          66          04  AC  C1 00009        ADDL3   LEN, (R6), DELTA                       ; 0701
                                                        6E  DD 0000E        PUSHL   DELTA                                  ; 0702
                        00000000G       00              01  FB 00010        CALLS   #1, SOR$$ALLOCATE                      ;
                                    04  AE              50  D0 00017        MOVL    R0, DELTA+4                            ;
                    04  BE          04  B6              66  28 0001B        MOVC3   (R6), @4(R6), @DELTA+4                 ; 0703
                        63          08  BC          04  AC  28 00021        MOVC3   LEN, @ADR, (R3)                        ;
                                            04  A6  9F 00027        PUSHAB  4(R6)                                          ; 0704
                                                        66  DD 0002A        PUSHL   (R6)                                   ;
                        00000000G       00              02  FB 0002C        CALLS   #2, SOR$$DEALLOCATE                    ;
                                        66              6E  7D 00033        MOVQ    DELTA, (R6)                            ; 0705
                                                            04 00036        RET                                            ; 0707

; Routine Size:  55 bytes,    Routine Base:  SOR$RO_CODE + 03A9

;  643          0708  2
;  644          0709  2      BIND
;  645          0710  2          DSC_ADR = VECTOR[CTX[COM_ROUTINES],0],
;  646          0711  2          DSC_LEN = VECTOR[CTX[COM_ROUTINES],1];
;  647          0712  2
;  648          0713  2      LOCAL
;  649          0714  2          ADJ_EQUAL,
;  650          0715  2          ADJ_COMPARE;
;  651          0716  2
;  652          0717  2
;  653          0718  2      ! Determine the longest output record length, COM_LRL_OUT.
;  654          0719  2      ! This also calculates COM_SPEC_TKS and COM_FORMATS.
;  655          0720  2      !
;  656          0721  2      CALC_LRL_OUT();
;  657          0722  2
;  658          0723  2
;  659          0724  2      ! See if we can use SOR$$KEY_SUB to generate the key comparison routines.
;  660          0725  2      ! We can do this if:
;  661          0726  2      !   There is only one record format,
;  662          0727  2      !   There are no conditional keys, and
;  663          0728  2      !   The data is simply the entire record (and not less than the LRL).
```

```
;  664     0729  2      !
;  665     0730  3      BEGIN LABEL LAB; LAB:
;  666     0731  4      BEGIN
;  667     0732  4      BUILTIN
;  668     0733  4          TESTBITSS,
;  669     0734  4          TESTBITCC;
;  670     0735  4      LOCAL
;  671     0736  4          HAVE_DATA,
;  672     0737  4          KEY_BUFF:       KEY_BLOCK,
;  673     0738  4          KFT_PTR:        REF KFT_TAB[];          ! Local pointer to KFT table
;  674     0739  4
;  675     0740  4      IF .CTX[COM_FORMATS] NEQ 1 THEN LEAVE LAB;
;  676     0741  4
;  677     0742  4      KFT_PTR = KFT[0,BASE_];
;  678     0743  4      HAVE_DATA = FALSE;
;  679     0744  4      KEY_BUFF[KEY_NUMBER] = 0;                    ! No keys yet
;  680     0745  4      DECR I FROM .CTX[COM_KFT_SIZ]-1 TO 0 DO
;  681     0746  5          BEGIN
;  682     0747  5          IF .KFT_PTR[0,KFT_C...A] THEN LEAVE LAB;
;  683     0748  5          IF .KFT_PTR[0,KFT_DATA]
;  684     0749  5          THEN
;  685     0750  6              BEGIN
;  686     0751  6              IF .KFT_PTR[0,KFT_CONSTANT] THEN LEAVE LAB;
;  687     0752  6              IF TESTBITSS(HAVE_DATA) THEN LEAVE LAB;
;  688     0753  6              IF .KFT_PTR[0,KFT_NDE_POS] NEQ 0 THEN LEAVE LAB;
;  689     0754  6              IF .KFT_PTR[0,KFT_NDE_SIZ] LSS .CTX[COM_LRL] THEN LEAVE LAB;
;  690     0755  6              END
;  691     0756  5          ELSE
;  692     0757  6              BEGIN
;  693     0758  6              LOCAL
;  694     0759  6                  FDT_PTR:REF FDT_TAB[1],
;  695     0760  6                  KBF:    REF KBF_BLOCK;
;  696     0761  6              KBF = KEY_BUFF[KEY_KBF(.KEY_BUFF[KEY_NUMBER])];
;  697     0762  6              FDT_PTR = FDT[.KFT_PTR[0,KFT_FDT_IDX],BASE_];
;  698     0763  6              KBF[KBF_TYPE] =     .FDT_PTR[0,FDT_TYPE];
;  699     0764  6              KBF[KBF_LENGTH] =   .FDT_PTR[0,FDT_FLD_SIZ];
;  700     0765  6              KBF[KBF_POSITION] = .FDT_PTR[0,FDT_FLD_POS];
;  701     0766  6              KBF[KBF_ORDER] =    .KFT_PTR[0,KFT_DESCEND];
;  702     0767  6              KEY_BUFF[KEY_NUMBER] = .KEY_BUFF[KEY_NUMBER] + 1;
;  703     0768  5              END;
;  704     0769  5          IF NOT .KFT_PTR[0,KFT_CONTINUE]
;  705     0770  5          THEN
;  706     0771  5              IF TESTBITCC(HAVE_DATA) THEN LEAVE LAB;
;  707     0772  5          KFT_PTR = KFT_PTR[1,BASE_];
;  708     0773  4          END;
;  709     0774  4      RETURN SOR$$KEY_SUB(KEY_BUFF[BASE_]);
;  710     0775  3      END;
;  711     0776  2      END;
;  712     0777  2
;  713     0778  2
;  714     0779  2      ! If we don't have the data, don't call user-written routines.
;  715     0780  2      !
;  716     0781  2      IF .CTX[COM_SORT_TYPE] NEQ TYP_K_RECORD
;  717     0782  2      THEN
;  718     0783  3          BEGIN
;  719     0784  3          IF .CTX[COM_COMPARE] NEQ 0 OR .CTX[COM_EQUAL] NEQ 0
;  720     0785  3          THEN
```

```
;  721      0786  3              RETURN SOR$$ERROR(SOR$_BAD_TYPE);
;  722      0787  2          END;
;  723      0788  2
;  724      0789  2
;  725      0790  2      ADJ_EQUAL = FALSE;
;  726      0791  2      ADJ_COMPARE = FALSE;
;  727      0792  2
;  728      0793  2
;  729      0794  2      ! If the user specified his own equal-key routine, call it.
;  730      0795  2      !
;  731      0796  3      BEGIN
;  732      0797  3      SWITCHES UNAMES;
;  733      0798  3      IF .CTX[COM_EQUAL] NEQ 0
;  734      0799  3      THEN
;  735      0800  4          BEGIN
;  736      0801  4          LOCAL
;  737      0802  4              TMP;
;  738      0803  4          TMP = .DSC_LEN;
;  739      0804  4          APPEND(.UE1[-1], UE1);
;  740      0805  4          APPEND(%UPVAL, CTX[COM_EQUAL]);
;  741      0806  4          APPEND(.UE2[-1], UE2);
;  742      0807  4          APPEND(%UPVAL, %REF(SOR$$ERROR));
;  743      0808  4          APPEND(.UE3[-1], UE3);
;  744      0809  4          CTX[COM_EQUAL] = .TMP;
;  745      0810  4          ADJ_EQUAL = TRUE;
;  746      0811  4          END
;  747      0812  3      ELIF .CTX[COM_NODUPS]
;  748      0813  3      THEN
;  749      0814  4          BEGIN
;  750      0815  4          ROUTINE NODUPS: JSB_EQUAL = SOR$_DELETE2;



              50 001C8111   8F  D0 00000 ;NODUPS
                                         U.1:      MOVL     #1868049, R0                       ; 0815
                                05 00007           RSB                                         ;

; Routine Size:  8 bytes,    Routine Base:  SOR$RO_CODE + 03E0


;  751      0816  4          CTX[COM_EQUAL] = NODUPS;
;  752      0817  4          END
;  753      0818  3      ELSE
;  754      0819  4          BEGIN
;  755      0820  4          !
;  756      0821  4          ! Leave COM_EQUAL equal to 0
;  757      0822  4          !
;  758      0823  4          0;
;  759      0824  3          END;
;  760      0825  2      END;
;  761      0826  2
;  762      0827  2
;  763      0828  2      ! Store the address of the length/address routine
;  764      0829  2      !
;  765      0830  3      BEGIN
```

```
;  766          0831  3     ROUTINE LENADR(S: REF VECTOR[,BYTE]; LEN, ADR): JSB_LENADR NOVALUE =
;  767          0832  4         BEGIN
;  768          0833  4         LEN = .(S[OFF_LEN])<0,16,0>;
;  769          0834  4         ADR = S[OFF_ADR];
;  770          0835  3         END;

                                        8A  DF 00000 LENADR:  PUSHAL  (R10)+                                  ; 0831
                              50    01  AA  3C 00002          MOVZWL  1(S), LEN                               ; 0833
                              5A        03  C0 00006          ADDL2   #3, ADR                                 ; 0834
                              51        5A  D0 00009          MOVL    R10, R1                                 ; 0835
                              5A        8E  D0 0000C          MOVL    (SP)+, R10                              ;
                                        05 0000F              RSB                                             ;

; Routine Size:  16 bytes,    Routine Base:  SOR$RO_CODE + 03E8

;  771          0836  3     CTX[COM_LENADR] = LENADR;
;  772          0837  3     END;
;  773          0838  2
;  774          0839  2
;  775          0840  2     ! If the user supplied a comparison routine, call it.
;  776          0841  2     !
;  777          0842  2     IF .CTX[COM_COMPARE] NEQ 0
;  778          0843  2     THEN
;  779          0844  3         BEGIN
;  780          0845  3         LOCAL
;  781          0846  3             TMP;
;  782          0847  3         TMP = .DSC_LEN;
;  783          0848  3         APPEND(.UE1[-1], UE1);
;  784          0849  3         APPEND(%UPVAL, CTX[COM_COMPARE]);
;  785          0850  3         APPEND(.UE4[-1], UE4);
;  786          0851  3         CTX[COM_COMPARE] = .TMP;
;  787          0852  3         ADJ_COMPARE = TRUE;
;  788          0853  3         END
;  789          0854  2     ELSE
;  790          0855  3         BEGIN
;  791          0856  3         CTX[COM_COMPARE] = COMPARE;
;  792          0857  3         END;
;  793          0858  2
;  794          0859  2     ! Store the address of the input reformatting routine
;  795          0860  2     !
;  796          0861  2     CTX[COM_INPUT] = INPUT;
;  797          0862  2
;  798          0863  2
;  799          0864  2     ! Store the length of an internal-format record
;  800          0865  2     !
;  801          0866  2     BEGIN
;  802          0867  2     LOCAL TMP;
;  803          0868  2     CTX[COM_LRL_INT] = TMP =
;  804          0869  3         OFF_ADR +                   ! Offset to start of the data
;  805          0870  3         .CTX[COM_LRL_OUT] +         ! The data
;  806          0871  3         .CTX[COM_SPEC_TKS];         ! The keys
;  807          0872  3     IF .TMP GTR MAX_REFSIZE
```

```
;  808          0873  3     THEN
;  809          0874  3         SOR$$ERROR(SOR$_SHR_BADLOGIC);  ! Not really bad logic, just rare.
;  810          0875  2     END;
;  811          0876  2
;  812          0877  2
;  813          0878  2     ! Adjust the actual addresses of the comparison and equal-key routines
;  814          0879  2     !
;  815          0880  2     IF .ADJ_EQUAL   THEN CTX[COM_EQUAL]   = .DSC_ADR + .CTX[COM_EQUAL];
;  816          0881  2     IF .ADJ_COMPARE THEN CTX[COM_COMPARE] = .DSC_ADR + .CTX[COM_COMPARE];
;  817          0882  2
;  818          0883  2
;  819          0884  2     ! Loop through the key field table, adjusting the positions of the fields
;  820          0885  2     ! within the internal format node.
;  821          0886  2     !
;  822          0887  2     DECR Z FROM .CTX[COM_KFT_SIZ]-1 TO 0 DO
;  823          0888  3         BEGIN
;  824          0889  3         LOCAL
;  825          0890  3             KFT_PTR: REF KFT_TAB[];             ! Local pointer to KFT table
;  826          0891  3         KFT_PTR = KFT[.Z,BASE ];                ! Pointer to key field entry
;  827          0892  3         IF .KFT_PTR[0, KFT_DATA]
;  828          0893  3         THEN
;  829          0894  3             KFT_PTR[0, KFT_NDE_POS] = .KFT_PTR[0, KFT_NDE_POS]
;  830          0895  3                 + OFF_ADR
;  831          0896  3         ELIF
;  832        P 0897  3             NOT ONEOF (.CTX[COM_SORT_TYPE],
;  833          0898  4                 BMSK_(TYP_K_ADDRESS,TYP_K_INDEX))
;  834          0899  3         THEN
;  835          0900  3             KFT_PTR[0, KFT_NDE_POS] = .KFT_PTR[0, KFT_NDE_POS]
;  836          0901  3                 + OFF_ADR + .CTX[COM_LRL_OUT]
;  837          0902  3         ELSE
;  838          0903  3             KFT_PTR[0, KFT_NDE_POS] = .KFT_PTR[0, KFT_NDE_POS]
;  839          0904  3                 + OFF_ADR
;  840          0905  2         END;
;  841          0906  2
;  842          0907  2     RETURN TRUE;
;  843          0908  1     END;


                                    07FC 00000          .ENTRY  SOR$$SPEC_KEY_SUB, Save R2,R3,R4,R5,R6,R7,- ; 0604
                                                                R8,R9,R10
                          5A     AC   AF  9E 00002      MOVAB   APPEND, R10                                 ;
                          5E   F800   CE  9E 00006      MOVAB   -2048(SP), SP                               ;
                          59   0108   CB  9E 0000B      MOVAB   264(CTX), R9                                ; 0661
                          58   0110   CB  9E 00010      MOVAB   272(CTX), R8                                ; 0662
                          57     18   AB  9E 00015      MOVAB   24(CTX), R7                                 ; 0710
                          56     1C   AB  9E 00019      MOVAB   28(CTX), R6                                 ; 0711
                    FEAF  CA           00  FB 0001D      CALLS   #0, CALC_LRL_OUT                            ; 0721
                          01   0083   CB  91 00022      CMPB    131(CTX), #1                                ; 0740
                                      70  12 00027      BNEQ    6$                                          ;
                          50           69  D0 00029      MOVL    (R9), KFT_PTR                               ; 0742
                                      55  D4 0002C      CLRL    HAVE_DATA                                   ; 0743
                                04    AE  B4 0002E      CLRW    KEY_BUFF                                    ; 0744
                          54   00FD   CB  9A 00031      MOVZBL  253(CTX), I                                 ; 0745
                                      53  11 00036      BRB     5$                                          ;
```

```
                                53        03    A0  9E 00038 1$:     MOVAB   3(KFT_PTR), R3                          ; 0747
                          59    63              03  E0 0003C         BBS     #3, (R3), 6$
                          16    63              06  E1 00040         BBC     #6, (R3), 2$                            ; 0748
                          51    63              01  E0 00044         BBS     #1, (R3), 6$                            ; 0751
                          4D    55              00  E2 00048         BBSS    #0, HAVE_DATA, 6$                       ; 0752
                                                60  B5 0004C         TSTW    (KFT_PTR)                               ; 0753
                                                49  12 0004E         BNEQ    6$
                                0084    CB      06  A0  B1 00050         CMPW    6(KFT_PTR), 132(CTX)                ; 0754
                                                29  1E 00056         BGEQU   3$
                                                3F  11 00058         BRB     6$
                                51              04  AE  3C 0005A 2$:     MOVZWL  KEY_BUFF, R1                        ; 0761
                                51              06 AE41 7E 0005E         MOVAQ   KEY_BUFF+2[R1], KBF
                                52              04  A0  9A 00063         MOVZBL  4(KFT_PTR), R2                      ; 0762
                                52                  06  C4 00067         MULL2   #6, R2
                                52                  68  C0 0006A         ADDL2   (R8), FDT_PTR
                                61                  62  9B 0006D         MOVZBW  (FDT_PTR), (KBF)                    ; 0763
                                      04        A1  02  A2  D0 00070         MOVL    2(FDT_PTR), 4(KBF)              ; 0765
                          52    63              01  05  EF 00075         EXTZV   #5, #1, (R3), R2                    ; 0766
                                      02        A1  52  B0 0007A         MOVW    R2, 2(KBF)
                                              04  AE  B6 0007E         INCW    KEY_BUFF                            ; 0767
                                              04  63  E8 00081 3$:     BLBS    (R3), 4$                            ; 0769
                          11                  55  00  E5 00084         BBCC    #0, HAVE_DATA, 6$                   ; 0771
                                              50  08  C0 00088 4$:     ADDL2   #8, KFT_PTR                         ; 0772
                                              AA  54  F4 0008B 5$:     SOBGEQ  I, 1$                               ; 0745
                                              04  AE  9F 0008E         PUSHAB  KEY_BUFF                            ; 0774
                                00000000G     00  01  FB 00091         CALLS   #1, SOR$$KEY_SUB
                                                  04  00098         RET
                                              01  58  AB  91 00099 6$:     CMPB    88(CTX), #1                     ; 0781
                                                  17  13 0009D         BEQL    8$
                                                  6B  D5 0009F         TSTL    (CTX)                               ; 0784
                                                  05  12 000A1         BNEQ    7$
                                              04  AB  D5 000A3         TSTL    4(CTX)
                                                  0E  13 000A6         BEQL    8$
                                    001C806C      8F  DD 000A8 7$:     PUSHL   #1867884                            ; 0786
                          00000000G 00            01  FB 000AE         CALLS   #1, SOR$$ERROR
                                                  04  000B5         RET
                                                  54  D4 000B6 8$:     CLRL    ADJ_EQUAL                           ; 0790
                                                  52  D4 000B8         CLRL    ADJ_COMPARE                         ; 0791
                                              04  AB  D5 000BA         TSTL    4(CTX)                              ; 0798
                                                  43  13 000BD         BEQL    9$
                                53                66  D0 000BF         MOVL    (R6), TMP                           ; 0803
                                        FE9B      CF  9F 000C2         PUSHAB  UE1                                 ; 0804
                                        FE93      CF  DD 000C6         PUSHL   UE1-4
                                6A                02  FB 000CA         CALLS   #2, APPEND
                                              04  AB  9F 000CD         PUSHAB  4(CTX)                              ; 0805
                                                  04  DD 000D0         PUSHL   #4
                                6A                02  FB 000D2         CALLS   #2, APPEND
                                        FEA4      CF  9F 000D5         PUSHAB  UE2                                 ; 0806
                                        FE9C      CF  DD 000D9         PUSHL   UE2-4
                                6A                02  FB 000DD         CALLS   #2, APPEND
                                6E 00000000G      00  9E 000E0         MOVAB   SOR$$ERROR, (SP)                    ; 0807
                                                  5E  DD 000E7         PUSHL   SP
                                                  04  DD 000E9         PUSHL   #4
                                6A                02  FB 000EB         CALLS   #2, APPEND
                                        FEA3      CF  9F 000EE         PUSHAB  UE3                                 ; 0808
                                        FE9B      CF  DD 000F2         PUSHL   UE3-4
                                6A                02  FB 000F6         CALLS   #2, APPEND
```

```
                        04  AB          53  D0 000F9          MOVL    TMP, 4(CTX)                                 ; 0809
                            54          01  D0 000FD          MOVL    #1, ADJ_EQUAL                               ; 0810
                                        0A  11 00100          BRB     10$                                         ; 0798
                    05      5B  AB      05  E1 00102  9$:     BBC     #5, 91(CTX), 10$                            ; 0812
                            04  AB  37   AA  9E 00107          MOVAB   NODUPS, 4(CTX)                              ; 0816
                            10  AB  3F   AA  9E 0010C  10$:    MOVAB   LENADR, 16(CTX)                             ; 0836
                                        6B  D5 00111          TSTL    (CTX)                                       ; 0842
                                        28  13 00113          BEQL    11$
                                53      66  D0 00115          MOVL    (R6), TMP                                   ; 0847
                                    FE45  CF  9F 00118          PUSHAB  UE1                                         ; 0848
                                    FE3D  CF  DD 0011C          PUSHL   UE1-4
                                6A          02  FB 00120          CALLS   #2, APPEND
                                            5B  DD 00123          PUSHL   CTX                                         ; 0849
                                            04  DD 00125          PUSHL   #4
                                6A          02  FB 00127          CALLS   #2, APPEND
                                    FE6F  CF  9F 0012A          PUSHAB  UE4                                         ; 0850
                                    FE67  CF  DD 0012E          PUSHL   UE4-4
                                6A          02  FB 00132          CALLS   #2, APPEND
                                6B          53  D0 00135          MOVL    TMP, (CTX)                                  ; 0851
                                52          01  D0 00138          MOVL    #1, ADJ_COMPARE                             ; 0852
                                            05  11 0013B          BRB     12$                                         ; 0842
                                6B  0000V   CF  9E 0013D  11$:    MOVAB   COMPARE, (CTX)                              ; 0856
                            08  AB  0000V   CF  9E 00142  12$:    MOVAB   INPUT, 8(CTX)                               ; 0861
                                53  0088    CB  9E 00148          MOVAB   136(CTX), R3                                ; 0868
                                50    02    A3  3C 0014D          MOVZWL  2(R3), R0                                   ; 0871
                                51    5E    AB  3C 00151          MOVZWL  94(CTX), R1
                                50          51  C0 00155          ADDL2   R1, R0
                                50          07  C0 00158          ADDL2   #7, TMP                                     ; 0870
                                63          50  B0 0015B          MOVW    TMP, (R3)                                   ; 0868
                        0000FFFF    8F      50  D1 0015E          CMPL    TMP, #65535                                 ; 0872
                                            0D  15 00165          BLEQ    13$                                         ; 0874
                                001C1124    8F  DD 00167          PUSHL   #1839396                                    ; 0874
                        00000000G   00          01  FB 0016D          CALLS   #1, SOR$$ERROR
                                04          54  E9 00174  13$:    BLBC    ADJ_EQUAL, 14$                              ; 0880
                            04  AB          67  C0 00177          ADDL2   (R7), 4(CTX)
                                03          52  E9 0017B  14$:    BLBC    ADJ_COMPARE, 15$                            ; 0881
                                6B          67  C0 0017E          ADDL2   (R7), (CTX)
                                51    00FD  CB  9A 00181  15$:    MOVZBL  253(CTX), Z                                 ; 0887
                                            28  11 00186          BRB     18$
                                50      00  B941  7E 00188  16$:    MOVAQ   @0(R9)[Z], KFT_PTR                          ; 0891
                        1B          03      A0  06 E0 0018D          BBS     #6, 3(KFT_PTR), 17$                         ; 0892
                        52  18000000    8F  58  AB 78 00192          ASHL    88(CTX), #402653184, R2                     ; 0898
                                            10  19 0019B          BLSS    17$
                                52          60  3C 0019D          MOVZWL  (KFT_PTR), R2                               ; 0901
                                54    02    A3  3C 001A0          MOVZWL  2(R3), R4
                                52          54  C0 001A4          ADDL2   R4, R2
                        60                  52      07  A1 001A7          ADDW3   #7, R2, (KFT_PTR)
                                            03  11 001AB          BRB     18$                                         ; 0900
                                60          07  A0 001AD  17$:    ADDW2   #7, (KFT_PTR)                               ; 0904
                                D5          51  F4 001B0  18$:    SOBGEQ  Z, 16$                                      ; 0892
                                50          01  D0 001B3          MOVL    #1, R0                                      ; 0907
                                            04  00 001B6          RET                                                 ; 0908

; Routine Size:  439 bytes,    Routine Base:  SOR$RO_CODE + 03F8
```

```
;  845     0909  1 ROUTINE INPUT
;  846     0910  1         (
;  847     0911  1         INPREC: REF VECTOR[2],          ! Length/address of input record
;  848     0912  1         OUTREC: REF VECTOR[,BYTE]       ! Area for reformatted output record
;  849     0913  1         ): JSB_INPUT =
;  850     0914  1 !++
;  851     0915  1 ! FUNCTIONAL DESCRIPTION:
;  852     0916  1 !
;  853     0917  1 !       Reformat an input record.
;  854     0918  1 !
;  855     0919  1 ! FORMAL PARAMETERS:
;  856     0920  1 !
;  857     0921  1 !       As described above
;  858     0922  1 !
;  859     0923  1 ! IMPLICIT INPUTS:
;  860     0924  1 !
;  861     0925  1 !       CTX             Longword pointing to work area (passed in COM_REG_CTX)
;  862     0926  1 !
;  863     0927  1 ! IMPLICIT OUTPUTS:
;  864     0928  1 !
;  865     0929  1 !       NONE
;  866     0930  1 !
;  867     0931  1 ! ROUTINE VALUE:
;  868     0932  1 !
;  869     0933  1 !       False iff the record should be dropped from the sort, true otherwise.
;  870     0934  1 !
;  871     0935  1 ! SIDE EFFECTS:
;  872     0936  1 !
;  873     0937  1 !       NONE
;  874     0938  1 !
;  875     0939  1 !--
;  876     0940  1 !
;  877     0941  2     BEGIN
;  878     0942  2     EXTERNAL REGISTER
;  879     0943  2         CTX =   COM_REG_CTX:    REF BLOCK[CTX_K_SIZE]
;  880     0944  2                                 FIELD(CTX_FIELDS);
;  881     0945  2     REGISTER
;  882     0946  2         CA =    COM_REG_CTX;
;  883     0947  2
;  884     0948  2     BIND
;  885     0949  2         RDT = CTX[COM_RDT_ADR]: REF RDT_TAB[],  ! Record definition table
;  886     0950  2         KFT = CTX[COM_KFT_ADR]: REF KFT_TAB[],  ! Key field table
;  887     0951  2         FDT = CTX[COM_FDT_ADR]: REF FDT_TAB[],  ! Field definition table
;  888     0952  2         CFT = CTX[COM_CFT_ADR]: REF CFT_TAB[];  ! Constant definition table
;  889     0953  2
;  890     0954  2     EXTERNAL ROUTINE
;  891     0955  2         SOR$$RDT:       CAL_CTXREG,
;  892     0956  2         SOR$$REFORM:    CAL_CTXREG;
;  893     0957  2
;  894     0958  2     LOCAL
;  895     0959  2         RDTPTR: REF RDT_TAB,
;  896     0960  2         KFT_IX,
;  897     0961  2         Z;
;  898     0962  2
;  899     0963  2     ! Determine the record type
;  900     0964  2     !
;  901     0965  2     Z = SOR$$RDT( INPREC[0], RDTPTR );
```

```
;  902          0966  2     SELECTONE .Z OF
;  903          0967  2         SET
;  904          0968  2         [0]:    RETURN FALSE;              ! omit the record
;  905          0969  2         [1]:    BEGIN
;  906          0970  3                 KFT_IX = .RDTPTR[0, RDT_KFT_IDX];
;  907          0971  3                 Z = SOR$$REFORM( INPREC[0], KFT[.KFT_IX,BASE_],
;  908          0972  3                     OUTREC[0], OUTREC[OFF_LEN]);
;  909          0973  3                 IF .Z NEQ 1 THEN (SOR$$ERROR(.Z); RETURN FALSE);
;  910          0974  3                 END;
;  911          0975  2         [OTHERWISE]:
;  912          0976  2                 (SOR$$ERROR(.Z); RETURN FALSE);
;  913          0977  2         TES;
;  914          0978  2
;  915          0979  2     (OUTREC[OFF_FMT])<0,8,0> = .KFT_IX;
;  916          0980  2
;  917          0981  2     IF NOT .CTX[COM_STABLE]
;  918          0982  2     THEN (OUTREC[OFF_STAB])<0,32,0> = 0
;  919          0983  2     ELIF .CTX[COM_MERGE]
;  920          0984  2     THEN (OUTREC[OFF_STAB])<0,32,0> = .CTX[COM_MRG_STREAM]
;  921          0985  2     ELSE (OUTREC[OFF_STAB])<0,32,0> = .CTX[COM_INP_RECNUM];
;  922          0986  2
;  923          0987  3     IF ONEOF_(.CTX[COM_SORT_TYPE], BMSK_(TYP_K_ADDRESS,TYP_K_INDEX))
;  924          0988  3     THEN
;  925          0989  3         BEGIN
;  926          0990  3         CH$MOVE(
;  927          0991  3             RAB$S_RFA,
;  928          0992  3             BLOCK[.CTX[COM_INP_CURR],DDB_RAB+RAB$W_RFA;,BYTE],
;  929          0993  3             OUTREC[OFF_ADR]);
;  930          0994  3         END;
;  931          0995  2
;  932          0996  2     RETURN TRUE;
;  933          0997  2
;  934          0998  1     END;


                                        .EXTRN  SOR$$RDT, SOR$$REFORM

                    5E          04  C2 00000 INPUT:  SUBL2   #4, SP                          ;  0909
                    53    0108  CB  9E 00003        MOVAB   264(CTX), R3                    ;  0950
                          4200  8F  BB 00008        PUSHR   #^M<R9,SP>                      ;  0965
         00000000G  00          02  FB 0000C        CALLS   #2, SOR$$RDT
                                50  D5 00013        TSTL    Z                               ;  0968
                                64  13 00015        BEQL    7$
                    01          50  D1 00017        CMPL    Z, #1                           ;  0969
                                1E  12 0001A        BNEQ    1$
                    51          6E  D0 0001C        MOVL    RDTPTR, R1                      ;  0970
                    52    04    A1  9A 0001F        MOVZBL  4(R1), KFT_IX
                          05    AA  9F 00023        PUSHAB  5(OUTREC)                       ;  0972
                                5A  DD 00026        PUSHL   OUTREC
                          00 B342 7F  00028        PUSHAQ  @0(R3)[KFT_IX]                  ;  0971
                                59  DD 0002C        PUSHL   INPREC                          ;  0972
         00000000G  00          04  FB 0002E        CALLS   #4, SOR$$REFORM
                    01          50  D1 00035        CMPL    Z, #1                           ;  0973
                                0B  13 00038        BEQL    2$
                                50  DD 0003A 1$:    PUSHL   Z                               ;  0976
         00000000G  00          01  FB 0003C        CALLS   #1, SOR$$ERROR
```

```
                                     36  11 00043       BRB     7$                                  ;
              04    AA               52  90 00045 2$:   MOVB    KFT_IX, 4(OUTREC)                   ; 0979
                    04        5B     AB  E8 00049       BLBS    91(CTX), 3$                         ; 0981
                                     6A  D4 0004D       CLRL    (OUTREC)                            ; 0982
                                     0F  11 0004F       BRB     5$                                  ;
                              5C     AB  95 00051 3$:   TSTB    92(CTX)                             ; 0983
                                     06  18 00054       BGEQ    4$                                  ;
                    6A        64     AB  D0 00056       MOVL    100(CTX), (OUTREC)                  ; 0984
                                     04  11 0005A       BRB     5$                                  ;
                    6A        7C     AB  D0 0005C 4$:   MOVL    124(CTX), (OUTREC)                  ; 0985
         50 18000000  8F      58     AB  78 00060 5$:   ASHL    88(CTX), #402653184, R0             ; 0987
                                     0B  18 00069       BGEQ    6$                                  ;
                    50      00A0     CB  D0 0006B       MOVL    160(CTX), R0                        ; 0992
       07    AA       24             A0  06  28 00070   MOVC3   #6, 36(R0), 7(OUTREC)               ; 0993
                    50               01  D0 00076 6$:   MOVL    #1, R0                              ; 0996
                                     02  11 00079       BRB     8$                                  ;
                                     50  D4 0007B 7$:   CLRL    R0                                  ; 0998
                    5E               04  C0 0007D 8$:   ADDL2   #4, SP                              ;
                                         05 00080       RSB                                         ;

; Routine Size:  129 bytes,    Routine Base:  SOR$RO_CODE + 05AF
```

```
;  936     0999  1 ROUTINE COMPARE
;  937     1000  1         (
;  938     1001  1         REC1:   REF VECTOR[,BYTE],      ! Address of internal format record
;  939     1002  1         REC2:   REF VECTOR[,BYTE]       ! Address of internal format record
;  940     1003  1         ): JSB_COMPARE =
;  941     1004  1 !++
;  942     1005  1 !
;  943     1006  1 ! FUNCTIONAL DESCRIPTION:
;  944     1007  1 !
;  945     1008  1 !       Compare records.
;  946     1009  1 !
;  947     1010  1 ! FORMAL PARAMETERS:
;  948     1011  1 !
;  949     1012  1 !       As described above
;  950     1013  1 !
;  951     1014  1 ! IMPLICIT INPUTS:
;  952     1015  1 !
;  953     1016  1 !       CTX             Longword pointing to work area (passed in COM_REG_CTX)
;  954     1017  1 !
;  955     1018  1 ! IMPLICIT OUTPUTS:
;  956     1019  1 !
;  957     1020  1 !       NONE
;  958     1021  1 !
;  959     1022  1 ! ROUTINE VALUE:
;  960     1023  1 !
;  961     1024  1 !       -1 if the first record collates before the second record
;  962     1025  1 !        0 if the records collate equal
;  963     1026  1 !        1 if the first record collates after the second record
;  964     1027  1 !
;  965     1028  1 ! SIDE EFFECTS:
;  966     1029  1 !
;  967     1030  1 !       NONE
;  968     1031  1 !
;  969     1032  1 !--
;  970     1033  2     BEGIN
;  971     1034  2     EXTERNAL REGISTER
;  972     1035  2         CTX =   COM_REG_CTX:    REF BLOCK[CTX_K_SIZE]
;  973     1036  2                                 FIELD(CTX_FIELDS);
;  974     1037  2     BIND
;  975     1038  2         RDT = CTX[COM_RDT_ADR]: REF RDT_TAB[],  ! Record definition table
;  976     1039  2         KFT = CTX[COM_KFT_ADR]: REF KFT_TAB[],  ! Key field table
;  977     1040  2         FDT = CTX[COM_FDT_ADR]: REF FDT_TAB[],  ! Field definition table
;  978     1041  2         CFT = CTX[COM_CFT_ADR]: REF CFT_TAB[];  ! Constant definition table
;  979     1042  2
;  980     1043  2     EXTERNAL ROUTINE
;  981     1044  2         SOR$$COMPARE:   CAL_CTXREG;             ! aka CA_LINKAGE
;  982     1045  2
;  983     1046  2     LOCAL
;  984     1047  2         KFT1:   REF KFT_TAB,
;  985     1048  2         KFT2:   REF KFT_TAB,
;  986     1049  2         EOK1,
;  987     1050  2         EOK2,
;  988     1051  2         S;
;  989     1052  2
;  990     1053  2     KFT1 = KFT[.REC1[OFF_FMT], BASE_];  ! Get 1st record's KFT pointer
;  991     1054  2     KFT2 = KFT[.REC2[OFF_FMT], BASE_];  ! Get 2nd record's KFT pointer
;  992     1055  2     EOK1 = FALSE;
```

```
;  993          1056  2      EOK2 = FALSE;
;  994          1057  2
;  995          1058  2      ! While there are more keys
;  996          1059  2      !
;  997          1060  2      WHILE TRUE DO
;  998          1061  3          BEGIN
;  999          1062  3          LOCAL
; 1000          1063  3              FLD1: VECTOR[2],    ! Length/address of field or constant
; 1001          1064  3              FLD2: VECTOR[2],    ! Length/address of field or constant
; 1002          1065  3              TYP1,
; 1003          1066  3              TYP2,
; 1004          1067  3              FDT_IX;             ! Index into FDT (or CFT) table
; 1005          1068  3
; 1006          1069  3          ! Advance both pointers to the next key description
; 1007          1070  3          !
; 1008          1071  3          WHILE 1 DO
; 1009          1072  4              BEGIN
; 1010          1073  4              IF NOT .KFT1[0,KFT_CONDX] THEN
; 1011          1074  4              IF NOT .KFT1[0,KFT_DATA] THEN EXITLOOP;
; 1012          1075  4              IF NOT .KFT1[0,KFT_CONTINUE] THEN (EOK1 = TRUE; EXITLOOP);
; 1013          1076  4              KFT1 = KFT1[1,BASE_];
; 1014          1077  3              END;
; 1015          1078  3          WHILE 1 DO
; 1016          1079  4              BEGIN
; 1017          1080  4              IF NOT .KFT2[0,KFT_CONDX] THEN
; 1018          1081  4              IF NOT .KFT2[0,KFT_DATA] THEN EXITLOOP;
; 1019          1082  4              IF NOT .KFT2[0,KFT_CONTINUE] THEN (EOK2 = TRUE; EXITLOOP);
; 1020          1083  4              KFT2 = KFT2[1,BASE_];
; 1021          1084  3              END;
; 1022          1085  3
; 1023          1086  3          ! The one that runs out of keys first collates less
; 1024          1087  3          !
; 1025          1088  3          IF (S = .EOK2 - .EOK1) NEQ 0 THEN RETURN .S;
; 1026          1089  3          IF .EOK1 THEN EXITLOOP;
; 1027          1090  3
; 1028          1091  3
; 1029          1092  3          FDT_IX = .KFT1[0,KFT_FDT_IDX];
; 1030          1093  3          IF .KFT1[0,KFT_CONSTANT]
; 1031          1094  3          THEN
; 1032          1095  4              BEGIN
; 1033          1096  4              TYP1 = DSC$K_DTYPE_Z;                   ! Unspecified
; 1034          1097  4              FLD1[0] = .KFT1[0, KFT_NDE_SIZ]
; 1035          1098  4              END
; 1036          1099  3          ELSE
; 1037          1100  4              BEGIN
; 1038          1101  4              TYP1 = .FDT[.FDT_IX, FDT_TYPE];
; 1039          1102  4              IF .TYP1 EQL DSC$K_DTYPE_P
; 1040          1103  4              THEN
; 1041          1104  4                  FLD1[0] = .FDT[.FDT_IX, FDT_FLD_SIZ]
; 1042          1105  4              ELSE
; 1043          1106  4                  FLD1[0] = .KFT1[0, KFT_NDE_SIZ]
; 1044          1107  3              END;
; 1045          1108  3          FLD1[1] = .KFT1[0,KFT_NDE_POS] + REC1[0];
; 1046          1109  3
; 1047          1110  3          FDT_IX = .KFT2[0,KFT_FDT_IDX];
; 1048          1111  3          IF .KFT2[0,KFT_CONSTANT]
; 1049          1112  3          THEN
```

```
;  1050          1113  4              BEGIN
;  1051          1114  4              TYP2 = .TYP1;                           ! Make it the same as the other
;  1052          1115  4              FLD2[0] = .KFT2[0, KFT_NDE_SIZ];
;  1053          1116  4              END
;  1054          1117  3          ELSE
;  1055          1118  4              BEGIN
;  1056          1119  4              TYP2 = .FDT[.FDT_IX, FDT_TYPE];
;  1057          1120  4              IF .TYP1 EQL DSC$K_DTYPE_Z THEN TYP1 = .TYP2;
;  1058          1121  4              IF .TYP2 EQL DSC$K_DTYPE_P
;  1059          1122  4              THEN
;  1060          1123  4                  FLD2[0] = .FDT[.FDT_IX, FDT_FLD_SIZ]
;  1061          1124  4              ELSE
;  1062          1125  4                  FLD2[0] = .KFT2[0, KFT_NDE_SIZ]
;  1063          1126  3              END;
;  1064          1127  3          FLD2[1] = .KFT2[0,KFT_NDE_POS] + REC2[0];
;  1065          1128  3
;  1066          1129  3          ! If the types are different, simply distinguish the records
;  1067          1130  3          !
;  1068          1131  3          IF (S = .TYP1 - .TYP2) NEQ 0 THEN RETURN SIGN(.S);
;  1069          1132  3
;  1070          1133  3          ! If different descending flags, the descending key comes first
;  1071          1134  3          !
;  1072          1135  3          IF (S = .KFT2[0,KFT_DESCEND]-.KFT1[0,KFT_DESCEND]) NEQ 0
;  1073          1136  3              THEN RETURN .S;
;  1074          1137  3
;  1075          1138  3          ! Finally, compare the fields
;  1076          1139  3          !
;  1077          1140  3          IF (S = SOR$$COMPARE(.TYP1, FLD1[0], FLD2[0])) NEQ 0 THEN
;  1078          1141  3              IF .KFT1[0,KFT_DESCEND] THEN RETURN -(.S) ELSE RETURN .S;
;  1079          1142  3
;  1080          1143  3              ! See whether this record definition is continued
;  1081          1144  3              ! Is this needed???
;  1082          1145  3              !
;  1083          1146  3              IF NOT .KFT1[0,KFT_CONTINUE] THEN EXITLOOP;
;  1084          1147  3              IF NOT .KFT2[0,KFT_CONTINUE] THEN EXITLOOP;
;  1085          1148  3
;  1086          1149  3              ! Advance to the next KFT entries
;  1087          1150  3              !
;  1088          1151  3              KFT1 = KFT1[1,BASE_];
;  1089          1152  3              KFT2 = KFT2[1,BASE_];
;  1090          1153  3
;  1091          1154  2              END;
;  1092          1155  2
;  1093          1156  2          ! The one that runs out of keys first collates less
;  1094          1157  2          !
;  1095          1158  2          IF (S = .KFT2[0,KFT_CONTINUE] - .KFT1[0,KFT_CONTINUE]) NEQ 0
;  1096          1159  2              THEN RETURN .S;
;  1097          1160  2
;  1098          1161  2          IF (S = .(REC1[OFF_STAB]) - .(REC2[OFF_STAB])) NEQ 0
;  1099          1162  2              THEN RETURN SIGN(.S);
;  1100          1163  2
;  1101          1164  2          RETURN 0;
;  1102          1165  1          END;


                                                        .EXTRN  SOR$$COMPARE
```

```
                5E          10  C2 00000 COMPARE:SUBL2   #16, SP                                    ;0999
                50      04  A9  9A 00003            MOVZBL  4(REC1), R0                            ;1053
                53  0108  DB40  7E 00007            MOVAQ   @264(CTX)[R0], KFT1
                50      04  AA  9A 0000D            MOVZBL  4(REC2), R0                            ;1054
                52  0108  DB40  7E 00011            MOVAQ   @264(CTX)[R0], KFT2
                       7E  7C 00017                 CLRQ    EOK2                                   ;1056
    05      03  A3          03  E0 00019 1$:        BBS     #3, 3(KFT1), 2$                        ;1073
    0F      03  A3          06  E1 0001E            BBC     #6, 3(KFT1), 4$                        ;1074
            06      03  A3      E8 00023 2$:        BLBS    3(KFT1), 3$                            ;1075
        04      AE          01  D0 00027            MOVL    #1, EOK1
                            05  11 0002B            BRB     4$
            53              08  C0 0002D 3$:        ADDL2   #8, KFT1                               ;1076
                            E7  11 00030            BRB     1$                                     ;1071
    05      03  A2          03  E0 00032 4$:        BBS     #3, 3(KFT2), 5$                        ;1080
    0E      03  A2          06  E1 00037            BBC     #6, 3(KFT2), 7$                        ;1081
            05      03  A2      E8 0003C 5$:        BLBS    3(KFT2), 6$                            ;1082
            6E              01  D0 00040            MOVL    #1, EOK2
                            05  11 00043            BRB     7$
            52              08  C0 00045 6$:        ADDL2   #8, KFT2                               ;1083
                            E8  11 00048            BRB     4$                                     ;1078
54          6E      04      AE  C3 0004A 7$:        SUBL3   EOK1, EOK2, S                          ;1088
                            03  13 0004F            BEQL    8$
                        00C9    31 00051            BRW     19$
            03      04      AE  E9 00054 8$:        BLBC    EOK1, 9$                               ;1089
                        00B1    31 00058            BRW     18$
            50      04      A3  9A 0005B 9$:        MOVZBL  4(KFT1), FDT_IX                        ;1092
04      03  A3              01  E1 0005F            BBC     #1, 3(KFT1), 10$                       ;1093
                            55  D4 00064            CLRL    TYP1                                   ;1096
                            18  11 00066            BRB     11$                                    ;1097
51          50              06  C5 00068 10$:       MULL3   #6, FDT_IX, R1                         ;1101
        51          0110    CB  C0 0006C            ADDL2   272(CTX), R1
                            61  9A 00071            MOVZBL  (R1), TYP1
                            15      55  D1 00074    CMPL    TYP1, #21                              ;1102
                            07      12 00077        BNEQ    11$
        10  AE      04          A1  3C 00079        MOVZWL  4(R1), FLD1                            ;1104
                            05  11 0007E            BRB     12$
        10  AE      06          A3  3C 00080 11$:   MOVZWL  6(KFT1), FLD1                          ;1106
                            63  3C 00085 12$:       MOVZWL  (KFT1), R1                             ;1108
14  AE                      59  C1 00088            ADDL3   REC1, R1, FLD1+4
            50      04          A2  9A 0008D        MOVZBL  4(KFT2), FDT_IX                        ;1110
    05      03  A2              01  E1 00091        BBC     #1, 3(KFT2), 13$                       ;1111
                            55  D0 00096            MOVL    TYP1, TYP2                             ;1114
                            1E  11 00099            BRB     15$                                    ;1115
            50              06  C4 0009B 13$:       MULL2   #6, R0                                 ;1119
            50      0110    CB  C0 0009E            ADDL2   272(CTX), R0
                            60  9A 000A3            MOVZBL  (R0), TYP2
                            55  D5 000A6            TSTL    TYP1                                   ;1120
                            03      12 000A8        BNEQ    14$
                            51  D0 000AA            MOVL    TYP2, TYP1
                            15      51  D1 000AD 14$:   CMPL    TYP2, #21                          ;1121
                            07      12 000B0        BNEQ    15$
        08  AE      04          A0  3C 000B2        MOVZWL  4(R0), FLD2                            ;1123
                            05  11 000B7            BRB     16$
        08  AE      06          A2  3C 000B9 15$:   MOVZWL  6(KFT2), FLD2                          ;1125
                            62  3C 000BE 16$:       MOVZWL  (KFT2), R0                             ;1127
0C  AE                      5A  C1 000C1            ADDL3   REC2, R0, FLD2+4
```

```
                54              55              51  C3 000C6          SUBL3   TYP2, TYP1, S                           ;1131
                                                5C  12 000CA          BNEQ    21$
          54    03   A2         01              05  EF 000CC          EXTZV   #5, #1, 3(KFT2), S                     ;1135
          50    03   A3         01              05  EF 000D2          EXTZV   #5, #1, 3(KFT1), R0
                                 54              50  C2 000D8          SUBL2   R0, S
                                                 40  12 000DB          BNEQ    19$
                                                 08  AE 9F 000DD       PUSHAB  FLD2                                ;1140
                                                 14  AE 9F 000E0       PUSHAB  FLD1
                                                 55  DD 000E3          PUSHL   TYP1
                00000000G        00              03  FB 000E5          CALLS   #3, SOR$$COMPARE
                                 54              50  D0 000EC          MOVL    R0, S
                                                 0A  13 000EF          BEQL    17$
                27        03     A3              05  E1 000F1          BBC     #5, 3(KFT1), 19$                    ;1141
                                 50              54  CE 000F6          MNEGL   S, R0
                                                 3E  11 000F9          BRB     23$
                                 0D        03    A3  E9 000FB 17$:     BLBC    3(KFT1), 18$                        ;1146
                                 09        03    A2  E9 000FF          BLBC    3(KFT2), 18$                        ;1147
                                 53              08  C0 00103          ADDL2   #8, KFT1                            ;1151
                                 52              08  C0 00106          ADDL2   #8, KFT2                            ;1152
                                                 FF0D 31 00109         BRW     1$                                  ;1060
          54    03   A2         01              00  EF 0010C 18$:     EXTZV   #0, #1, 3(KFT2), S                  ;1158
          50    03   A3         01              00  EF 00112          EXTZV   #0, #1, 3(KFT1), R0
                                 54              50  C2 00118          SUBL2   R0, S
                                                 05  13 0011B          BEQL    20$
                                 50              54  D0 0011D 19$:     MOVL    S, R0                               ;1159
                                                 17  11 00120          BRB     23$
                                 54              69  6A C3 00122 20$:  SUBL3   (REC2), (REC1), S                  ;1161
                                                 0F  13 00126          BEQL    22$
                                                 54  D5 00128 21$:     TSTL    S                                   ;1162
                                                 50  DC 0012A          MOVPSL  R0
          50                     50              02  02  EF 0012C      EXTZV   #2, #2, R0, R0
                                 50              01  50  C3 00131      SUBL3   R0, #1, R0
                                                 02  11 00135          BRB     23$
                                                 50  D4 00137 22$:     CLRL    R0                                  ;1164
                                 5E              18  C0 00139 23$:     ADDL2   #24, SP                             ;1165
                                                 05  0013C            RSB

; Routine Size:  317 bytes,    Routine Base:  SOR$RO_CODE + 0630
```

```
;  1104          1166  1 GLOBAL ROUTINE SOR$$COMPATIBLE(
;  1105          1167  1         KFT1:   REF KFT_TAB,    ! Address of KFT entry for first key
;  1106          1168  1         KFT2:   REF KFT_TAB     ! Address of KFT entry for second key
;  1107          1169  1         ): CAL_CTXREG =
;  1108          1170  1 !++
;  1109          1171  1 !
;  1110          1172  1 ! FUNCTIONAL DESCRIPTION:
;  1111          1173  1 !
;  1112          1174  1 !       Determine whether keys are compatible.
;  1113          1175  1 !
;  1114          1176  1 ! FORMAL PARAMETERS:
;  1115          1177  1 !
;  1116          1178  1 !       As described above
;  1117          1179  1 !
;  1118          1180  1 ! IMPLICIT INPUTS:
;  1119          1181  1 !
;  1120          1182  1 !       CTX             Longword pointing to work area (passed in COM_REG_CTX)
;  1121          1183  1 !
;  1122          1184  1 ! IMPLICIT OUTPUTS:
;  1123          1185  1 !
;  1124          1186  1 !       NONE
;  1125          1187  1 !
;  1126          1188  1 ! ROUTINE VALUE:
;  1127          1189  1 !
;  1128          1190  1 !        0 if the keys are compatible.
;  1129          1191  1 !       -1 if the keys are incompatible with KFT1 coming first.
;  1130          1192  1 !        1 if the keys are incompatible with KFT2 coming first.
;  1131          1193  1 !
;  1132          1194  1 ! SIDE EFFECTS:
;  1133          1195  1 !
;  1134          1196  1 !       NONE
;  1135          1197  1 !
;  1136          1198  1 !--
;  1137          1199  2     BEGIN
;  1138          1200  2     EXTERNAL REGISTER
;  1139          1201  2         CTX =   COM_REG_CTX:    REF BLOCK[CTX_K_SIZE]
;  1140          1202  2                                 FIELD(CTX_FIELDS);
;  1141          1203  2     BIND
;  1142          1204  2         FDT = CTX[COM_FDT_ADR]: REF FDT_TAB[];  ! Field definition table
;  1143          1205  2
;  1144          1206  2     LOCAL
;  1145          1207  2         FDT_IX,
;  1146          1208  2         FLD1_TYP:       BYTE,
;  1147          1209  2         FLD2_TYP:       BYTE,
;  1148          1210  2         FLD1_LEN:       WORD,
;  1149          1211  2         FLD2_LEN:       WORD,
;  1150          1212  2         FLD1_SCA:       BYTE,
;  1151          1213  2         FLD2_SCA:       BYTE,
;  1152          1214  2         S;
;  1153          1215  2
;  1154          1216  2     FDT_IX = .KFT1[0,KFT_FDT_IDX];
;  1155          1217  2     IF .KFT1[0,KFT_CONSTANT]
;  1156          1218  2     THEN
;  1157          1219  3         BEGIN
;  1158          1220  3         FLD1_TYP = DSC$K_DTYPE_Z;
;  1159          1221  3         FLD1_LEN = .KFT1[0, KFT_NDE_SIZ];
;  1160          1222  3         FLD1_SCA = 0;
```

```
;  1161          1223  3          END
;  1162          1224  2      ELSE
;  1163          1225  3          BEGIN
;  1164          1226  3          FLD1_TYP = .FDT[.FDT_IX, FDT_TYPE];
;  1165          1227  3          IF .FLD1_TYP EQL DSC$K_DTYPE_P
;  1166          1228  3          THEN
;  1167          1229  3              FLD1_LEN = .FDT[.FDT_IX, FDT_FLD_SIZ]
;  1168          1230  3          ELSE
;  1169          1231  3              FLD1_LEN = .KFT1[0, KFT_NDE_SIZ];
;  1170          1232  3          FLD1_SCA = .FDT[.FDT_IX, FDT_SCALE];
;  1171          1233  2          END;
;  1172          1234  2
;  1173          1235  2      FDT_IX = .KFT2[0,KFT_FDT_IDX];
;  1174          1236  2      IF .KFT2[0,KFT_CONSTANT]
;  1175          1237  2      THEN
;  1176          1238  3          BEGIN
;  1177          1239  3          FLD2_TYP = .FLD1_TYP;
;  1178          1240  3          FLD2_LEN = .KFT2[0, KFT_NDE_SIZ];
;  1179          1241  3          FLD2_SCA = 0;
;  1180          1242  3          END
;  1181          1243  2      ELSE
;  1182          1244  3          BEGIN
;  1183          1245  3          FLD2_TYP = .FDT[.FDT_IX, FDT_TYPE];
;  1184          1246  3          IF .FLD1_TYP EQL DSC$K_DTYPE_Z THEN FLD1_TYP = .FLD2_TYP;
;  1185          1247  3          IF .FLD2_TYP EQL DSC$K_DTYPE_P
;  1186          1248  3          THEN
;  1187          1249  3              FLD2_LEN = .FDT[.FDT_IX, FDT_FLD_SIZ]
;  1188          1250  3          ELSE
;  1189          1251  3              FLD2_LEN = .KFT2[0, KFT_NDE_SIZ];
;  1190          1252  3          FLD2_SCA = .FDT[.FDT_IX, FDT_SCALE];
;  1191          1253  2          END;
;  1192          1254  2
;  1193          1255  2
;  1194          1256  2      ! If the types are different, simply distinguish the records
;  1195          1257  2      !
;  1196          1258  2      IF (S = .FLD1_TYP - .FLD2_TYP) NEQ 0 THEN RETURN SIGN(.S);
;  1197          1259  2
;  1198          1260  2
;  1199          1261  2      ! Check the lengths
;  1200          1262  2      !
;  1201          1263  2      IF .FLD1_TYP NEQ DSC$K_DTYPE_T AND .FLD1_TYP NEQ DSC$K_DTYPE_Z
;  1202          1264  2      THEN
;  1203          1265  2          IF (S = .FLD1_LEN - .FLD2_LEN) NEQ 0 THEN RETURN SIGN(.S);
;  1204          1266  2
;  1205          1267  2
;  1206          1268  2      ! Check the scales
;  1207          1269  2      !
;  1208          1270  2      IF (S = .FLD1_SCA - .FLD2_SCA) NEQ 0 THEN RETURN SIGN(.S);
;  1209          1271  2
;  1210          1272  2
;  1211          1273  2      ! If different descending flags, the descending key comes first
;  1212          1274  2      !
;  1213          1275  2      IF (S = .KFT2[0,KFT_DESCEND]-.KFT1[0,KFT_DESCEND]) NEQ 0
;  1214          1276  2          THEN RETURN .S;
;  1215          1277  2
;  1216          1278  2
;  1217          1279  2      ! The fields are compatible
```

```
; 1218        1280  2    !
; 1219        1281  2    RETURN 0;
; 1220        1282  1    END;



                                    03FC 00000        .ENTRY  SOR$$COMPATIBLE, Save R2,R3,R4,R5,R6,R7,R8,-;  1166
                                                              R9                                          ;
                   53       04  AC   D0 00002        MOVL    KFT1, R3                                    ;  1216
                   50       04  A3   9A 00006        MOVZBL  4(R3), FDT_IX                               ;
      0A      03   A3           01   E1 0000A        BBC     #1, 3(R3), 1$                               ;  1217
                                54   94 0000F        CLRB    FLD1_TYP                                    ;  1220
                   57       06  A3   B0 00011        MOVW    6(R3), FLD1_LEN                             ;  1221
                                58   94 00015        CLRB    FLD1_SCA                                    ;  1222
                                1F   11 00017        BRB     4$                                          ;  1217
      51           50           06   C5 00019 1$:    MULL3   #6, FDT_IX, R1                              ;  1226
                   51     0110  CB   C0 0001D        ADDL2   272(CTX), R1                                ;
                   54           61   90 00022        MOVB    (R1), FLD1_TYP                              ;
                   15           54   91 00025        CMPB    FLD1_TYP, #21                               ;  1227
                                06   12 00028        BNEQ    2$                                          ;
                   57       04  A1   B0 0002A        MOVW    4(R1), FLD1_LEN                             ;  1229
                                04   11 0002E        BRB     3$                                          ;
                   57       06  A3   B0 00030 2$:    MOVW    6(R3), FLD1_LEN                             ;  1231
                   58       01  A1   90 00034 3$:    MOVB    1(R1), FLD1_SCA                             ;  1232
                   52       08  AC   D0 00038 4$:    MOVL    KFT2, R2                                    ;  1235
                   50       04  A2   9A 0003C        MOVZBL  4(R2), FDT_IX                               ;
      0B      03   A2           01   E1 00040        BBC     #1, 3(R2), 5$                               ;  1236
                   51           54   90 00045        MOVB    FLD1_TYP, FLD2_TYP                          ;  1239
                   55       06  A2   B0 00048        MOVW    6(R2), FLD2_LEN                             ;  1240
                                56   94 0004C        CLRB    FLD2_SCA                                    ;  1241
                                25   11 0004E        BRB     9$                                          ;  1236
                   50           06   C4 00050 5$:    MULL2   #6, R0                                      ;  1245
                   50     0110  CB   C0 00053        ADDL2   272(CTX), R0                                ;
                   51           60   90 00058        MOVB    (R0), FLD2_TYP                              ;
                                54   95 0005B        TSTB    FLD1_TYP                                    ;  1246
                                03   12 0005D        BNEQ    6$                                          ;
                   54           51   90 0005F        MOVB    FLD2_TYP, FLD1_TYP                          ;
                   15           51   91 00062 6$:    CMPB    FLD2_TYP, #21                               ;  1247
                                06   12 00065        BNEQ    7$                                          ;
                   55       04  A0   B0 00067        MOVW    4(R0), FLD2_LEN                             ;  1249
                                04   11 0006B        BRB     8$                                          ;
                   55       06  A2   B0 0006D 7$:    MOVW    6(R2), FLD2_LEN                             ;  1251
                   56       01  A0   90 00071 8$:    MOVB    1(R0), FLD2_SCA                             ;  1252
                   50           54   9A 00075 9$:    MOVZBL  FLD1_TYP, S                                 ;  1258
                   59           51   9A 00078        MOVZBL  FLD2_TYP, R9                                ;
                   50           59   C2 0007B        SUBL2   R9, S                                       ;
                                1F   12 0007E        BNEQ    11$                                         ;
      0E                        54   91 00080        CMPB    FLD1_TYP, #14                               ;  1263
                                0F   13 00083        BEQL    10$                                         ;
                                54   95 00085        TSTB    FLD1_TYP                                    ;
                                0B   13 00087        BEQL    10$                                         ;
                   50           57   3C 00089        MOVZWL  FLD1_LEN, S                                 ;  1265
                   51           55   3C 0008C        MOVZWL  FLD2_LEN, R1                                ;
                   50           51   C2 0008F        SUBL2   R1, S                                       ;
                                0B   12 00092        BNEQ    11$                                         ;
```

```
                        50          58  9A 00094 10$:    MOVZBL  FLD1_SCA, S                     ; 1270
                        51          56  9A 00097         MOVZBL  FLD2_SCA, R1                    ;
                        50          51  C2 0009A         SUBL2   R1, S                           ;
                                    11  13 0009D         BEQL    12$                             ;
                                    50  D5 0009F 11$:     TSTL    S                               ;
                                    51  DC 000A1         MOVPSL  R1                              ;
          51            51        02  02  EF 000A3         EXTZV   #2, #2, R1, R1                  ;
                        51        01  51  C3 000A8         SUBL3   R1, #1, R1                      ;
                        50        01  51  D0 000AC         MOVL    R1, R0                          ;
                                    04  000AF             RET                                     ;
          50    03    A2          01  05  EF 000B0 12$:     EXTZV   #5, #1, 3(R2), S                ; 1275
          51    03    A3          01  05  EF 000B6         EXTZV   #5, #1, 3(R3), R1               ;
                        50          51  C2 000BC         SUBL2   R1, S                           ;
                                    02  12 000BF         BNEQ    13$                             ;
                                    50  D4 000C1         CLRL    R0                              ; 1281
                                    04  000C3 13$:     RET                                     ; 1282

; Routine Size:  196 bytes,    Routine Base:  SOR$RO_CODE + 076D
```

```
1 ROUTINE CLEAN_UP: CAL_CTXREG NOVALUE =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Release resources allocated by this module.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !
1 !       NONE (signals errors)
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
2     BEGIN
2     EXTERNAL REGISTER
2         CTX =   COM_REG_CTX:     REF CTX_BLOCK;
2     IF .CTX[COM_WRK_ADR] NEQ 0 AND .CTX[COM_WRK_END] NEQ 0
2     THEN
3         BEGIN
3         CTX[COM_WRK_ADR] = .CTX[COM_WRK_END] - .CTX[COM_WRK_SIZ];
3         SOR$$DEALLOCATE(.CTX[COM_WRK_SIZ], CTX[COM_WRK_ADR]);
2         END;
1     END;
```

```
                                    0000 00000 CLEAN_UP:
                                                       .WORD   Save nothing                    ; 1283
                 50     0128    CB   9E 00002          MOVAB   296(CTX), R0                    ; 1315
                                60   D5 00007          TSTL    (R0)                            ;
                                1B   13 00009          BEQL    1$                              ;
                        012C    CB   D5 0000B          TSTL    300(CTX)                        ;
                                15   13 0000F          BEQL    1$                              ;
       60    012C   CB  0124    CB   C3 00011          SUBL3   292(CTX), 300(CTX), (R0)        ; 1318
                                50   DD 00019          PUSHL   R0                              ; 1319
                        0124    CB   DD 0001B          PUSHL   292(CTX)                        ;
        00000000G   00          02   FB 0001F          CALLS   #2, SOR$$DEALLOCATE             ;
                                     04 00026 1$:      RET                                     ; 1321
```

```
; Routine Size:  39 bytes,    Routine Base:  SOR$RO_CODE + 0831

; 1261          1322  1
; 1262          1323  1 END
; 1263          1324  0 ELUDOM



;                                  PSECT SUMMARY

;       Name                      Bytes                          Attributes

;  SOR$RO_CODE--------------------2      4  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  SOR$RO_CODE                        2136  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  .  ABS  .                             0  NOVEC,NOWRT,NORD ,NOEXE,NOSHR,  LCL,  ABS,  CON,NOPIC,ALIGN(0)



;                           Library Statistics

;                                        -------- Symbols --------      Pages       Processing
;       File                             Total    Loaded   Percent     Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1       9776       109         1        581       00:01.0
;  _$255$DUA28:[SORT32.SRC]SORLIB.L32;1     409       151        36         34       00:00.4
;  _$255$DUA28:[SORT32.SRC]SRTSPC.L32;1     120        20        16         12       00:00.1
;  _$255$DUA28:[SORT32.SRC]OPCODES.L32;1    343        15         4         18       00:00.4



;                                  COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:SORSPEC/OBJ=OBJ$:SORSPEC MSRC$:SORSPEC/UPDATE=(ENH$:SORSPEC)

; Size:         2047 code + 93 data bytes
; Run Time:       00:45.3
; Elapsed Time:   02:32.2
; Lines/CPU Min:   1754
; Lexemes/CPU-Min: 27070
; Memory Used:  262 pages
; Compilation Complete
```

SRTSPCBLI
LIS

SORSPEC
LIS

SORTRNVEC
LIS

SORSORT
LIS

SORSPCUTI
LIS

SRTSPC
LIS